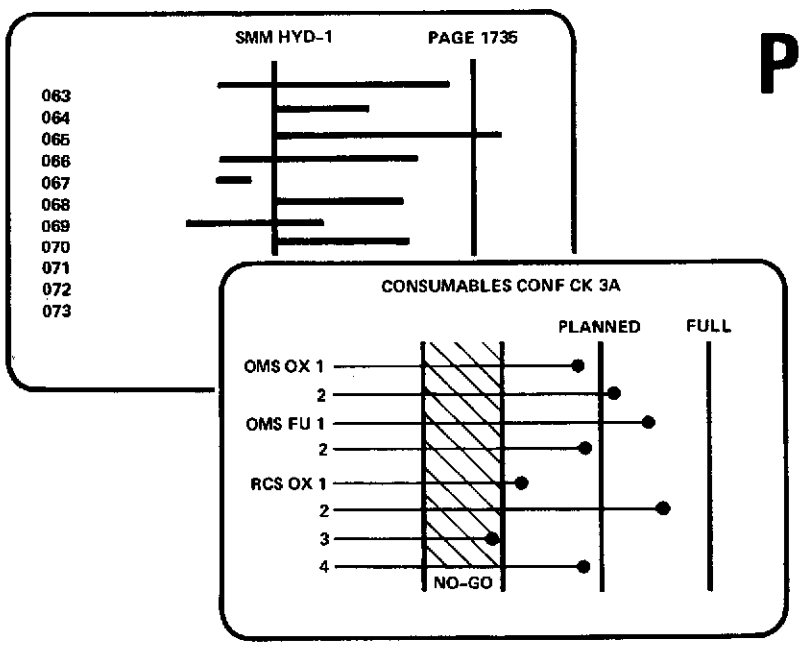
DRL NO. T-959
LINE ITEM 4
DRD NO. MA-129T

9330-75-005
FEBRUARY 1975
NASA CR-
141759

# FINAL REPORT-
# STUDY OF SPACE SHUTTLE ORBITER SYSTEM MANAGEMENT COMPUTER FUNCTION



## VOLUME II
# AUTOMATED PERFORMANCE VERIFICATION CONCEPTS






Prepared for
LYNDON B. JOHNSON
SPACE CENTER, NASA
HOUSTON, TEXAS

Prepared by
ADVANCED PROGRAMS DEPARTMENT
UNDER CONTRACT NUMBER
NAS 9-13887

HARRIS
COMMUNICATIONS AND INFORMATION HANDLING

HARRIS CORPORATION Electronic Systems Division*
P.O. Box 37, Melbourne, Florida 32901 305/727-4000
*(formerly RADIATION)

9330-75-005
FEBRUARY 1975



FINAL REPORT
STUDY OF SPACE SHUTTLE ORBITER
SYSTEM MANAGEMENT COMPUTER FUNCTION

# AUTOMATED PERFORMANCE VERIFICATION CONCEPTS

**VOLUME II**

HARRIS CORPORATION, ELECTRONIC SYSTEMS DIVISION

Prepared for
LYNDON B. JOHNSON SPACE CENTER, NASA
HOUSTON, TEXAS
Under Contract Number NAS 9-13887

# PREFACE

This volume of the Study Final Report consolidates the findings of investigations on concepts and techniques in automated performance verification. The investigations were undertaken to provide additional insight into the design methodology and to develop a consolidated technology base from which to analyze performance verification design approaches.

As the title implies, the bulk of this volume is dedicated to relating concepts. Communication has been given priority over writing form, so the presentation throughout the volume is informal. The material has been developed with digital processor implementation in mind. It has, however, also been kept as general as possible so that it may contribute to the widest possible applications.

Much of the material is based on a design handbook prepared for the Kennedy Space Center which addresses a related problem and should be referenced for additional detail.*

In view of the Orbiter SM design evolution, a distinction between performance monitoring and performance verification might be appropriate here. The former connotes the presentation of performance parameters, whereas the latter, performance verification, connotes the presentation of whether these parameters are acceptable or not. The latter is a more significant operation in that it includes decision regarding operational integrity. This volume addresses performance verification, and in particular, the problem of presenting functional failure information.

*Handbook, Design of Automated Redundancy Verification, Ford Hasslinger, Moreno, July 1971, Radiation, Inc. 7620-71-001, NTIS No. CR 125311.

TABLE OF CONTENTS,

Volume II of 2

TABLE OF CONTENTS (Continued)

# GLOSSARY

False Alarm — A performance verification error committed when an IBV is declared unacceptable when it is actually acceptable.

IBV — Acronym for Item Being Verified. Associated with each IBV is a single verification instance and status indication.

Measure of Performance — A verification signal which represents how well an IBV is performing.

Miss — A performance verification error committed when an unacceptable IBV is not declared as such within a predefined notification time.

MOP — Acronym for Measure of Performance.

Notification Time — The time interval between the point where an IBV actually becomes unacceptable and the point where the performance verification system declares this condition.

Performance Monitoring — The process of displaying performance measures of principal system elements.

Performance Verification — The process of determining whether a principal system element is performing satisfactorily. Performance Verification implies a decision process.

Performance Verification System — The system which accomplishes performance verification.

Post-Decision Smoother — A smoothing mechanism which makes status decisions beased on consecutive decisions from a preceding performance measure limit checker.

Predecision Smoother — A smoothing mechanism which operates on a performance measure prior to a status decision. For computer applications, these will typically be recursive filters.

Principal System — The system whose operation is being verified. The system which contains equipment whose status is to be identified.

PVS — Acronym for Performance Verification System.

Redundancy — The capacity of having more than one way to accomplish the same function where the alternative methods may assume the function within a prescribed time. So long as the alternative method is acceptable (in some sense), there is no implication of equivalent capability.

Smoothing — The process of deleting anomalies or transients from an information stream. In the context of Performance Verification, smoothing is the process used to decrease false alarm status decision.

Status — A qualitative and usually broad statement regarding the operational integrity of the item under contention, e.g., good, marginal, poor; go/no-go.

Status Resolution — The process of identifying the correct IBV status when several related IBV's all indicate an unacceptable condition due to failure in the one IBV.

# 1.0 INTRODUCTION

Design of performance verification systems, like the weather, is much talked about and little done about. With increasing frequency, vestiges of designs crop up here and there — especially in the aircraft industry. The demand (or need) is obviously increasing. Unfortunately, many of these designs have been approached as special problems to be addressed after all other design has been completed — or in some cases is actually in the prototype stages. The designs almost always are realized due to the dedicated efforts of cockpit designers and avionics designers who custom fit each indication, solving each problem as it comes up. This is a reflection, not on the designers, but rather on the tools with which they have to work. Why is performance verification a problem? Haven't all the rough spots been worked out with experience gained in such systems as the Apollo Automatic Checkout, the Navy's VAST and numerous other checkout systems? With few exceptions, such designs either verify logical sequences of events or operate on a stimulus-response basis for performance testing. They do not make real-time decisions regarding performance while the item being checked is actually performing its function. Designs incorporating real-time performance verification on an automated basis are just coming of age. Probably the field which contains the most precedence is that of real-time process control. This field has been doing performance verification for years. The purpose of this volume is to capitalize on portions of the process control and related disciplines to structure a general theory of performance verification. This is accomplished by advancing a design approach as well as design criteria. General application of real-time performance verification to a large, sophisticated system demands more than happenstance design. It demands a formal approach. Designers must be able to assess and control risks. They must be able to identify what is to be verified.

In order that the material will be of maximum, immediate benefit, a verification process has been structured. An overview of this process and the verification problem at large is provided in Sections 2.0 and 3.0. Section 4.0 describes a design structure for accomplishing a verification design. Specific components of the methodology are detailed in Sections 5.0 through 8.0. Summary and Conclusions appear in Section 9.0.

The developed methodology is applied in an example using the Orbiter Hydraulics System. This example is contained in Appendix A. Appendix B contains detailed discussions of performance verification techniques.

## 2.0 WHAT IS PERFORMANCE VERIFICATION?

It is appropriate to begin by defining performance verification. As used in this report, performance verification is the process of determining the operational integrity, in a timely manner, of specific system functions with emphasis on this determination while the function is performing mission operations. To advance the definition a step further, the term operational integrity implies knowledge of more than just the value of a function variable, e.g., 40 pounds of thrust. Rather, it implies a measure be inherent to verification which indicates how well the function is performing. Unfortunately, achieving this measure is only half the story. Of immediate interest to the crew (and to related functions) is whether the function is performing satisfactorily or not. When a few independent functions are involved, performance measures can be displayed and the crew can determine if they are operating "good enough." As functional relationships increase and the number of functions grows, this procedure quickly places a sizable burden on the crew. The solution; automate. Thus, performance verification will also contain a judgment process which decides whether a performance measure is acceptable or not. This discrete performance indication is termed status and, as will become evident in subsequent discussions, has the added advantage of logical manipulation.

To recap, the performance verification process determines status (acceptable, unacceptable operation) of specified functions by making quantitative decisions regarding the value of a function performance measure (or measures). Arriving at a suitable performance measure is seldom easy and implementing this measure can become complex. These are the first contributors to cost and complexity. There are more. Functions often change operating modes with time. This typically implies additional performance measures be defined and developed. Functions are not always exhaustively exercised throughout a mission, which implies that only portions of the function are checked — especially if nonlinearities are involved. If this is insufficient, added sophistication will be required.

As indicated, performance verification is a decision process. Mankind has not yet devised a method of providing decision-makers with flawless information on which to base the decision. Where decisions are made, errors will also be made. And, therein lies the risk which is assumed when automated performance verification is implemented. In the final analysis the systems designer is trading off timely information about the state of his system, against the possibility that this information may not always be correct. Stated another way, he is trading off ignorance of system state, or the operator burden of reducing this ignorance, against an automated but

imperfect state determination. Possibility of error is being exchanged for timely information and/or reduced operator burden. Objectively, information errors are the price paid for the added information. Is the price worth it? This fundamental question should be asked about each instance of verification contemplated. What are the consequences of manual verification or waiting until the crew discovers a failed function? — for they will, with certainty, eventually do so.

There is, however, one additional aspect of operator failure detection which must be considered in the tradeoff. This is the necessity of the operator to resolve ambiguities which may be present. Symptoms, not causes, are observed by an operator. If subsequent operator corrective action depends on identifying the specific function which failed, ambiguous symptoms will require on-the-spot troubleshooting to effect this identity. Thus, this troubleshooting must be made simple and/or the symptoms unambiguous.

Redundancy design too, can affect the decision of whether to use automated verification. If switching is used to change on-line elements in a redundant set and further, if the function demands a rapid recovery rate, automated verification will be essential.

## 3.0 APPROACHING THE VERIFICATION PROBLEM

What is involved in the automated verification of performance? Certainly it is more than testing a collection of system measurements for limit values. It may not be practical to set a limit on a measurement (independently) or to attach a functional or physical meaning to the test results. What about false alarms or misses? What about erroneous outputs from a failed function causing subsequent "good" functions to appear failed? How are the results to be presented? How much interpretation is required on the part of the crew? What specific functions are to be verified? How are these selected? How are off-line functions to be handled? These are but a few of the questions which arise on first examination of the problem. The material presented in this volume is intended to provide a structured approach to the design of an automated performance verification system. The approach is one of breaking the problem into manageable pieces which are recognized as disciplines in their own right. Design considerations are then provided for each piece. Synthesis of the pieces is treated in Section 4.0 and is woven as a theme throughout the volume.

The first and major cleaving of the problem is keyed to the contrast between those things which characterize the principal system and those things which characterize the design of the performance verification system.* This is portrayed in Figure 3.0. The former establishes what is to be verified and the nature of this verification. It provides design requirements for the verification design. The latter is the design realization which meets these requirements. Section 3.1 below discusses the principal system characterization and 3.2, performance verification design. Sections 7.0 and 5.0 address these subjects in detail.

### 3.1 Principal System Characterization

What is the nature of the thing to be verified and what are the verification requirements? A partial answer to this question is to be found in the conceptual system documentation. Here are established mission profiles, requirements definitions, operations requirements/policies, design philosophies, user interfaces, and broad specifications of the overall system as seen by the user. From these documents can be extracted requirements definitions for the verification system. In the case of the Orbiter mission profile, SM is seen to

*So that the performance verification system can be distinguished from that which it is verifying, the latter is termed the principal system.

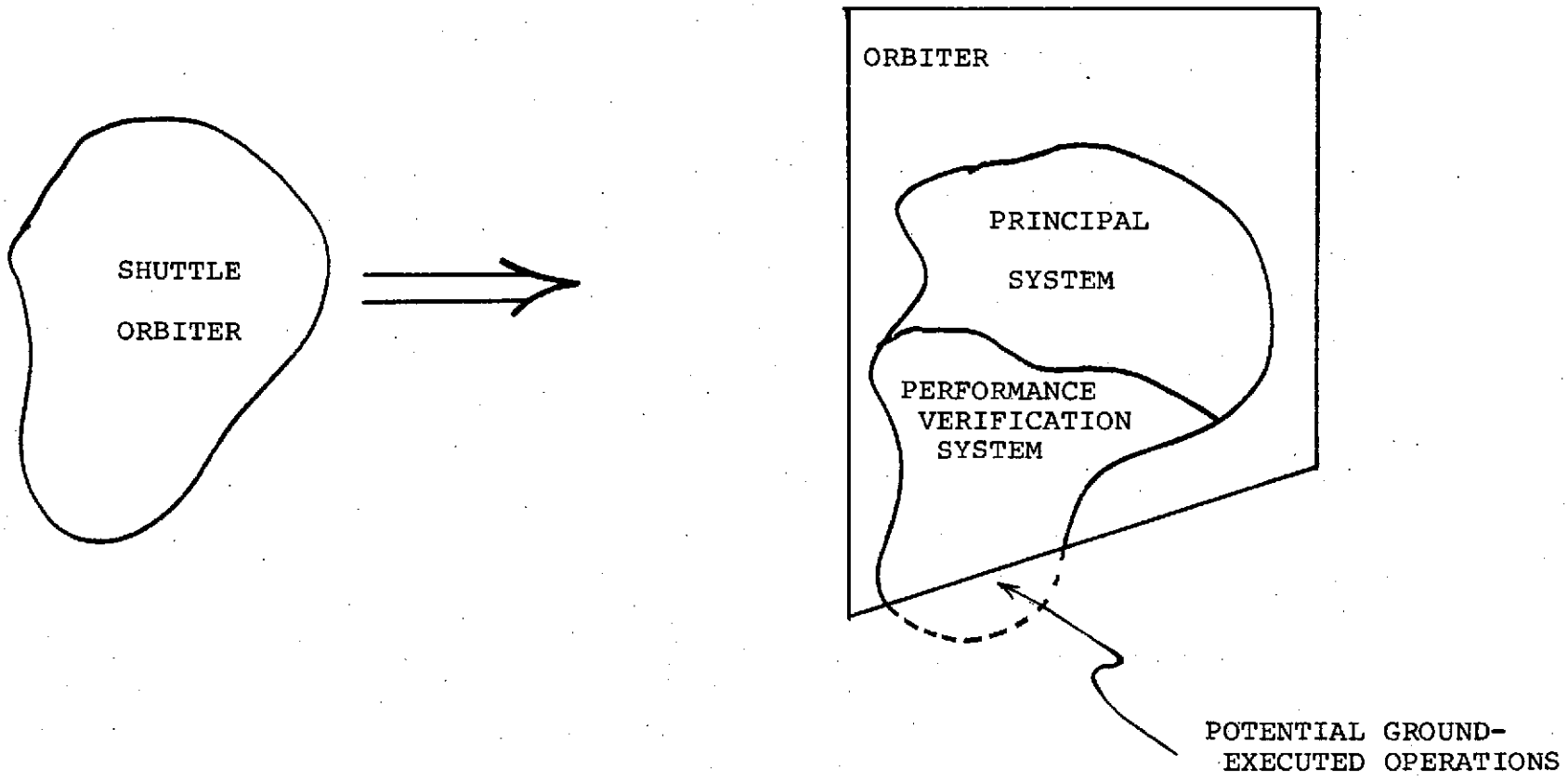



Figure 3.0. The First Partition

operate in a minimum of three potential modes corresponding to preflight, flight and post-flight. It is quite likely that verification interests will differ for each mode. In fact, this interest will likely vary for the various phases during flight. Thus, SM will be required to adapt to mission phase. The concept of SM evolution through Orbiter development phases also establishes a flexibility and growth requirement on the applications programs, computer time line and the operating system. Descriptions of principal system functions will provide:

a. Response time requirements of the performance verification system (PVS), i.e., how long it takes to react,

b. Gross requirements of PVS accuracy or confidence (false alarm and miss tolerance),

c. Definitive operating modes under which functions must be verified,

d. How frequently a function must be verified and when,

e. Performance criteria on which to base measures of operational integrity.

The above enumeration is a good checklist for verification design data. In any well conceived system, much of the necessary information already exists and need only be reformulated in terms of PVS requirements.

We have thus far established a source of PVS performance requirements which provide the initial definition of, and constraints on, the design. The system is beginning to take form. It now remains to identify what in the principal system is to be verified. There are two obvious extremes to this question. One is to display status of every subassembly on board the Orbiter. The other is a single display which indicates whether the Orbiter as a whole is operative or not. The final solution will without question lie between these extremes. The former represents an unnecessarily complex system and presents an unmanageable data resolution problem on the crew. In addition, it has been indicated that performance verification is inherently a decision process. The large number of decisions exemplified by the former approach can only increase the overall probability of committing verification errors. The latter approach may be acceptable under some particular circumstances, such as just prior to lift-off. It rarely, however, provides enough information for the crew to make an intelligent decision regarding alternative courses of

action. A single status indication for a complex system rarely makes sense when applied throughout the mission. A system GO for launch is different than that for injection and still different from that of orbital extraction. A system as complex as the Orbiter employs different functions and different operating modes as a mission progresses. The context of system GO changes. Finally, it is virtually impossible to determine a single system measure of performance on which to base system status. Such indications are inevitably made up of combinations of numerous smaller status indications, e.g., countdown checklists.

How then does one decide on what is to be verified and constrain the quantity of these items to a "reasonable" number? It is imperative that what is being verified be specifically identified. Is it the entire S-band communications or just an S-band transmitter? This identifies to the crew exactly what portion of the system the status display represents. This specific identification is also the only way that performance criteria can be established and requirements can be apportioned to the verification system. The specific items are called Items Being Verified (IBV's) and each represents an independent instance of verification with its associated independent status. This division or partitioning of the principal system is depicted in Figure 3.1.

There is an optimum (at least in a qualitative sense) IBV set for each system. Section 7.0 provides details on ways of arriving at this optimum and these will not be pursued here. For reasons of cost, complexity and decision errors, the number of IBV's should be quite small (equating to large functions being verified). Also contributing to verification of large functions is the level of status significance. The IBV should be as large as the significance or consequence of failure. There is little point in displaying the status of a nonredundant receiver IF, for when it fails, so does the entire receiver. Here, no corrective action can be taken to recover from failure of a piece of the receiver, so status of the entire receiver is more meaningful and requires less interpretation on the part of the crew. This example points out two other considerations in IBV determination. First, redundancy (or ability to repair) displays a key role in the determination. The presence of redundancy always represents a recovery mechanism and the institution of recovery is a principal reason for verification. Also, the crew will be vitally interested in their chances of completing a mission phase prior to committing to that phase. The amount of redundancy present is a direct measure of their potential success.

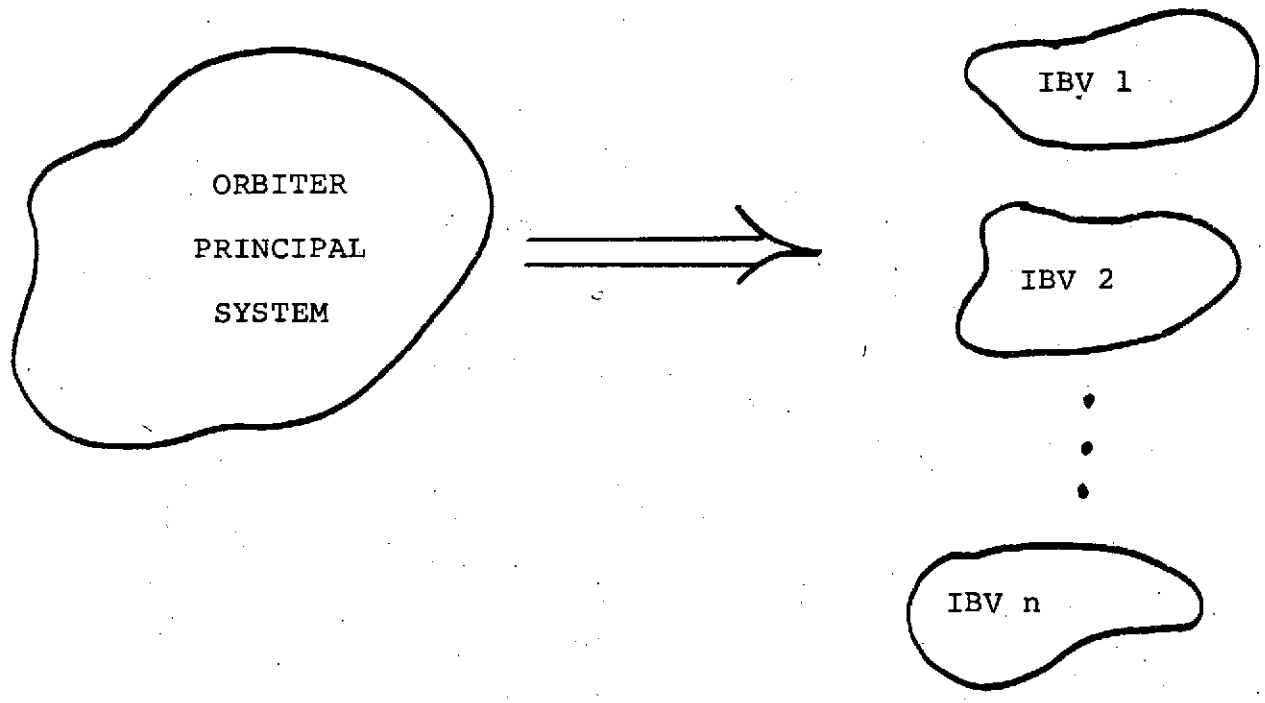



Figure 3.1. The Second Partition

The second consideration brought out by the above example is that the IBV's should be at the same level of significance as that for which the crew can do something about a failure, i.e., the level of recovery. In the example, the crew could do nothing about the failure of the receiver IF but can recover from receiver failure by using a redundant unit.* IF status is not consistent with recovery level. If the IBV is lower than the recovery level, extraneous information is introduced. If it is larger than recovery level, ambiguities will result. And, this leads directly into the considerations which motivate increasing the number of IBV's, i.e., implementing (for the same system) smaller IBV's. For any system as complex as the Orbiter, graceful (hopefully) degradation will begin to occur as functions fail. The system's capabilities (or potential capabilities) begin to deteriorate albeit the system at large is still functioning "acceptably." IBV's should be sufficiently small to allow an accurate portrayal of this deterioration. They should be of an equivalent "granularity" to the equipment degradation process. For example, each mass memory aboard the Orbiter should be an IBV since failure of one does not constitute total loss of the mass memory function. The operating system (OS) has more than a casual interest in the status of its resources. When a mass memory fails, the OS will undoubtedly resort to new storage/retrieval algorithms and an adjusted set of priorities. This IBV selection is in consonance with the degradation granularity and, accordingly, provides unambiguous information upon which recovery can act directly. An alternative approach would be to treat the entire mass memory function as an IBV. In principal, this would be a perfectly acceptable approach. It would, however, not be consistent with the degradation granularity — especially as recovery must account for equipment within an IBV thus defined.

Extrapolating the above results, one may conclude that all redundancy should be verified. There is obviously a practical limit to how far down into the circuitry verification should go. In general, real-time verification of redundant P.C. cards is seldom warranted — especially if this redundancy is self-recovering, i.e., hardwired. Aside from the complexity this introduces, the efficacy of the verification is limited since, likely, the reliability of equipment at this level will be such that there is a very low probability of failure during the mission. This is as it should be for unverified equipment. If there is not a low probability of failure for all such

*An extension of this conclusion relates to deferred maintenance. Unless there are compelling reasons to the contrary, IBV's should not be established to facilitate ground repair after flight.

unverified redundancy taken as a whole, the redundancy design should be seriously questioned. The preceding discussion should by no means be construed to imply that some redundancy will never be verified. All redundancy should be verified during preflight checkout. The task of keeping tabs on this redundancy once checkout is completed, however, should not in all cases be a function of performance verification.

Before departing this topic, there is a final point which should be made. Section 7.0 contains a discussion of status deduction wherein status of equipment within IBV's can sometimes be deduced based on logical combinations of alternative tests and IBV status indications. We will not dwell on this subject here. Suffice it to say that there are ways of determining equipment status without actually going to the expense of directly verifying it. This in effect extends the power of a single instance of verification.

## 3.2 The Verification Process

The preceding section indicated where the performance verification system requirements originate, the implication of these requirements and how IBV's are selected and identified. This section will describe how the instance of verification associated with each IBV thus identified is constructed. A detailed discussion of the material presented here in capsule form will be found in Section 5.0.

What operations must be performed for automated performance verification? Let us begin by recalling (Section 2.0) what automated performance verification is.... the process of determining the operational integrity, in a timely manner, of specific system functions with emphasis on this determination while the function is performing mission operations. With the added material in Section 3.1, we can now be more specific and state that it is the process of determining status of an IBV in a timely manner by making quantitative decisions regarding the value of its performance measure (or measures). This is depicted in Figure 3.2-1 as a third level of partitioning. Two operations are obvious from the definition alone: performance measure extraction and decision making. In the former, the word, extraction, has not been arbitrarily chosen. Once an IBV performance measure has been selected, it is quite likely that the process will involve more than a direct comparison of a transducer value with a stored limit, especially in the discrete time domain of a digital computer. To better understand this statement, let us examine some general performance measures, what implementation restrictions they impose and how the process can be handled.

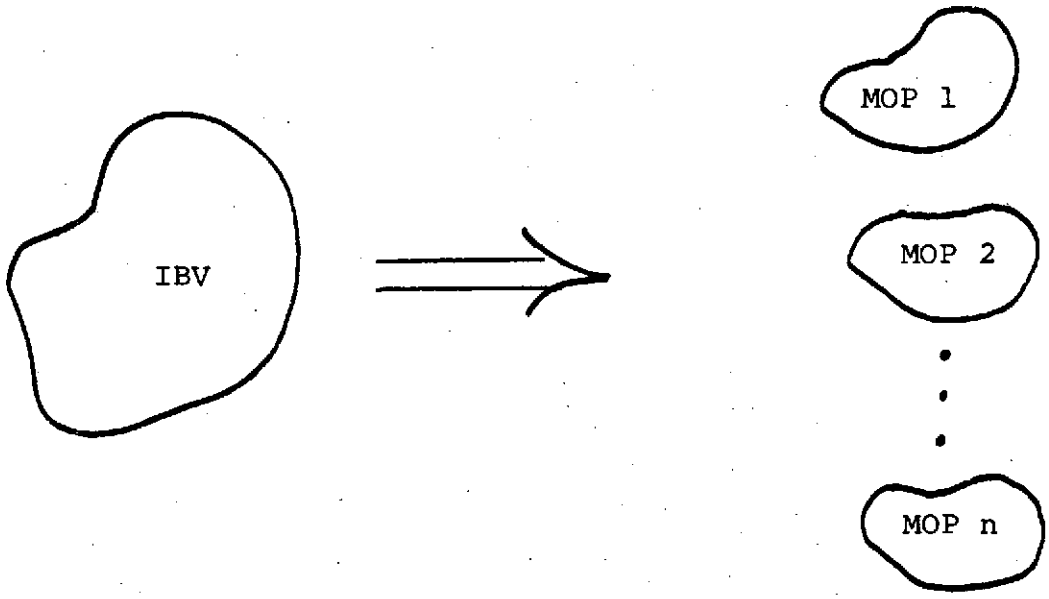



Figure 3.2-1. The Third Partition

Performance measures are quite sensitive to the type of function being measured and, except for implementation realization, have no theoretical bounds. Realization restrictions are:

a. The measure must be quantifiable, i.e., one must be able to describe it by numbers. This is seldom a problem for most physical measures unless they become complex, consisting of a loosely coupled combination of element measures. For example, one can identify dependability as a performance measure for a communication system. This has intuitive appeal but obviously is so broad that any numbering scheme assigned would be arbitrary. One could further qualify dependability and state as a performance measure: the percent of error-free messages transacted which were delayed no more than one minute. This is quantifiable.

b. One must have access to the constituent properties and combine them in a consistent and logical manner. Said another way, one must be able to extract the basic properties (or dimensions) of the measure and construct the measure of performance from these properties. In the communications system example above, the rules for combining the basic properties are clear. Achieving the basic properties, viz., number of messages with errors, total number of messages transacted and delay of each message, is feasible but non-trivial. Assuming one can get electrical connection (or software linkage) to the proper places, each property can be accumulated.

c. One must be able to state the range of values which constitute acceptable or satisfactory operation. A performance measure, or measures, must satisfy the crew that it is an adequate indicator of IBV operational integrity. Furthermore, they must be satisfied that the threshold set for acceptable operation is a suitable indication of when they can depend on the IBV. In the communications example one must be able to state the percent of error-free messages not delayed more than one minute, below which system operation will be considered unacceptable.

d. The measure must be that which is most representative of the function being performed by the IBV. Rate of liquid oxygen being pumped is a measure

of rocket engine performance but this indicator alone would not satisfy most people as a sole performance measure. What does a rocket engine do? It provides a minimum thrust along a prescribed vector at a predetermined time. The thrust vector and timing can be readily relegated to navigation and guidance functions, leaving minimum thrust as the respresentative performance measure. An "almost direct" method of thrust indication can be achieved by chamber pressure measurements or gimbal strain measurements.

e. The performance measure must be such that it is responsive, yet not volatile. By responsive, we mean that the measure must lend itself to being implemented such that a status change will be displayed with minimum delay (from when the change actually occurred). An implementation can be considered responsive if the status (change) delay approaches the natural period of the IBV. By volatile is meant the implementation is susceptible to transient conditions, i.e., hypersensitive. The manifestation of this condition is false alarms. The manifestation of sluggish (nonresponsive) operation is misses.

This restriction introduces dynamics into the verification problem and it is dynamics which contrast real-time performance verification with automatic checkout systems. The latter is strictly a static system. The performance measure for the communications system example in (a) above is not quite complete, for it contains no dynamical restrictions. Should the percent of error-free messages be calculated over a daily (24 hr.) basis, an hourly basis or instantaneously? Since the nature and density of the traffic typically varies during a day, there will be a difference in these approaches. In effect, we are asking the duration over which an averaged sample should be taken. Selecting a daily basis will provide a good indication of operational integrity but if performed literally will result in very sluggish reporting of status change. This can be speeded up considerably by placing the restriction of a daily moving average which is determined, say, every 15 minutes. An instantaneous approach would be quite responsive,

but would also be volatile. In addition, the variation of message density may require such an approach to be adaptive.

This problem cannot be solved here for its solution depends on operations restrictions on the system. The example, however, should point out the necessity of dynamic considerations.

The above realization restrictions should provide guidelines in selecting performance measures and should point out the difficulties which can be encountered. Real-time performance verification is, inescapably, a statistical problem involving sample sizes, averaging, goodness of fit, correlation, decision mechanisms (hypothesis testing), etc.

Implicit to the above restrictions is the necessity of extracting from the IBV the most information for minimum cost. And, by "most information" we mean that information which comes the closest to satisfying the performance measure and which relates performance of the IBV as a whole. In the realm of nuts and bolts this information must be extracted from IBV signals, where a signal is any modulated physical property, not necessarily an electrical waveform. Furthermore, IBV output signals will, with few exceptions, come the closest to describing the IBV performance as a whole. For, it is the output of a device which comes closest to completely describing its function. From an implementation standpoint, one must structure a performance measure signal (although it will actually be a sequence of numerical representations in the case of a digital computer) from information contained in IBV output signals. This information may not always be in the form desired of a performance measure and consequently require manipulation. We saw an example of this earlier where the number of error messages were tabulated and, in effect, averaged over a period of time. Averaging is certainly not the only operation that can be performed on the signals (or data). A navigation system might be interested in Root-Mean-Squared error. Other systems may require the performance measure be expressed as Integral-Squared error. In fact, any of the statistical or probabilistic operations appearing in the literature is a reasonable candidate. The key lies in the concise measure of performance selected and in our ability to make definitive statements regarding what constitutes acceptable and unacceptable operation of an IBV.

In expressing performance measures which meet the realization restrictions cited earlier, one is often confronted with the inability to make a definitive statement regarding acceptable operation without first making reference to another IBV or to that IBV input. Assuming the performance measure

were, simply, output volts, it would be nice if one could state the range of acceptable operation (decision rule) in terms of this output voltage alone. For example, the decision rule could be,

$$-4.0 \leq \text{output volts} \leq 8.2 \Rightarrow \text{acceptable,}$$
$$\text{unacceptable otherwise.}$$

Unfortunately, this is not always the case. It may be that a definitive statement about IBV performance can only be made with respect to another IBV or with respect to that IBV's input. (Note that, in the final analysis, status can only be decided upon using definitive statements about performance. The performance criteria, must be manipulated such that definitive statements can be made.) An example of the former might be that an IBV will be considered acceptable if its output is within 0.6 volt of a second, identical IBV. Two redundant navigation systems might be considered operative if the RMS difference between their normal reckoned positions remained within 7.0 meters over a one-hour period. In the case of output with respect to input, consider a voltage amplifier. The performance measure could easily be the ratio of output to input volts. The decision rule could then be:

$$9X\ (\text{input volts}) \leq \text{output} \Rightarrow \text{acceptable,}$$
$$\text{unacceptable otherwise.}$$

It is worthy to note that the decision rules used to judge IBV operation will likely represent a dividing line between very good performance under ideal conditions and good-to-marginal performance. Under emergency conditions, the crew will likely be willing to accept much worse performance. A second, emergency set of decision rules may be well advised to reassess orbiter status for marginal conditions when trouble arises.

From all that has been said about performance measures and IBV's, the reader may have the impression that each IBV will have a single measure of performance. Can an IBV have more than one performance measure? Yes. In fact, it is often easier to handle several, ideally independent, measures separately and establish a decision rule for each. This is effectively constraining the problem to canonical or elemental form and makes the dimensions easier to cope with. One may not be able to state a decision rule about a single, complex measure but he can often attach physical and operational significance to its constituent parts. When several performance measures are treated in this way, an algorithmic relationship can be established to relate the status of the several measures, i.e., good-bad, to that of their IBV. For example, an IBV may have the following performance measures

and decision rules for each of the measures:

$3.6 \leq$ peak output volts $\leq 4.8$ $\Rightarrow$ acceptable
kHz $4.0 \leq$ frequency $\leq 4.6$ kHz $\Rightarrow$ acceptable
phase $\leq 20^{o}$ $\Rightarrow$ acceptable

These rules establish the status definition for each performance measure. It can then be stated that the IBV status will be considered unacceptable if at least one of the above measures is unacceptable. This process amounts to synthesizing Measure of Performance (MOP) status indications into IBV status and is depicted in Figure 3.2-2. Note it is the reverse of that shown in Figure 3.2-1.

To this point we have discussed the defining, structuring and implementation of performance measures and status development. We are now ready to discuss a final point which has been touched upon frequently and is a direct outgrowth of real-time dynamics. This point is errors. Performmance verification can commit two types of errors. It can indicate an IBV has failed when, indeed, it has not — a false alarm. It can also fail to indicate, within a prescribed time limit, that an IBV has failed when, indeed, it has — a miss. The time limit constraint in the definition of a miss is crucial. If a status change is not indicated in a timely manner, in many cases it may as well not have been indicated at all. Except for hardware failure within the performance verification system, the probability of virtually any design ultimately indicating an IBV failure will approach unity — if one waits long enough. Furthermore, the crew itself, will also ultimately, detect the failure. If response time is not a consideration, one can seriously question the need for real-time verification. The remaining virtues are reduction of crew workload and possible isolation information. Under these circumstances such verification could be performed as point checks when mission phases change, perhaps as part of configuration checks.

Of the two types of errors, false alarms usually cause the most concern. We shall sketch the avoidance of false alarms. A detailed discussion will be found in Section 8.0.

False alarms occur due to two principal causes: hardware/software failure within the performance verification system and anomalies in the developed performance measure signal which are misinterpreted by the status decision mechanism. We shall address the latter, control of the former being amply treated throughout the literature.

For effective control, the nature of the signal anomalies should be understood. These can be inherent to the

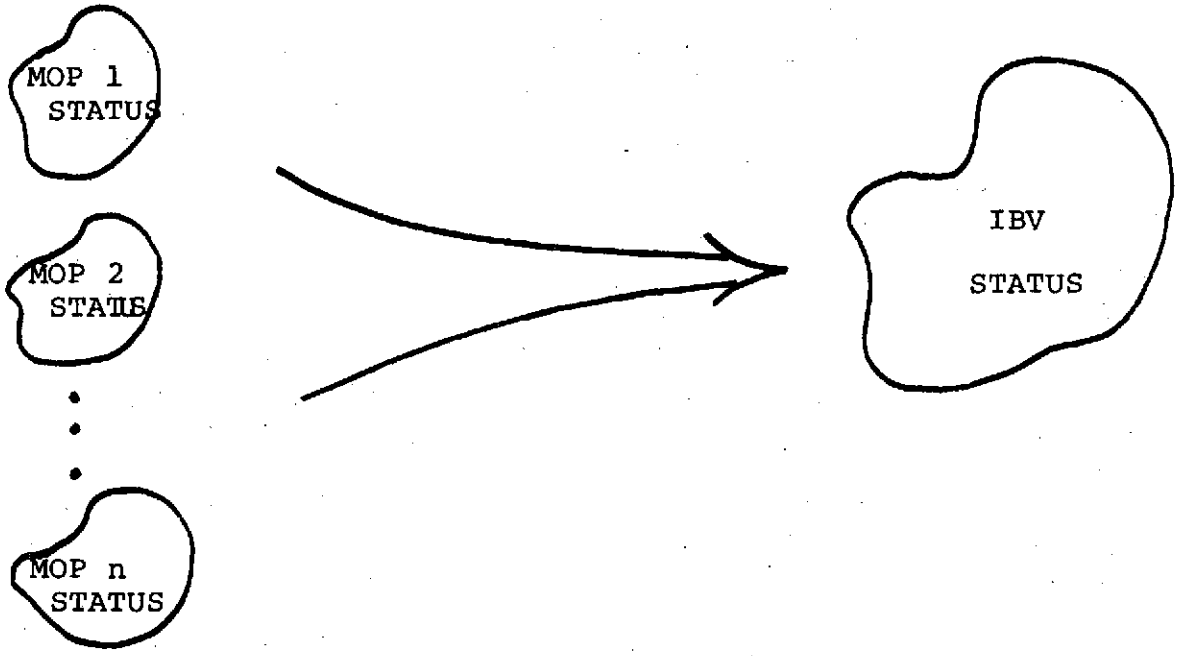



Figure 3.2-2. Synthesizing Measure of Performance Status Into IBV Status

IBV signals in the form of overshoot, transients, crosstalk, noise, hum, timing/phase deviations, race, etc. A digitally-implemented performance verification system will introduce, in addition to some of those mentioned, anomalies arising from truncation, roundoff and finite word length. Control of the latter will reside almost exclusively within the software numerical design. A classic example of this is setting of the integration interval for a numerical integration problem. Figure 3.2-3 illustrates what happens to the error as the interval is made progressively smaller. Error increase past the minimum is caused by truncation errors as the number of iterations increases. Control of numerical errors is a study in its own right and is quite sensitive to machine architecture and the nature of the function being processed. Suffice it to say that error compensation cannot begin unless:

a. The errors produce an invariant bias which is known and can be compensated for, and/or

b. The errors do not consistently accumulate in one direction, i.e., they alternate in sign.

The goal of error control is to make the IBV status indications insensitive to all anomalies. Our signal processing colleagues would call this a filtering problem and we shall adopt their descriptive designation. We shall limit our discussion to filters which reduce random effects (as opposed to special application filters which compensate for bias or discrete portions of the frequency spectrum). Such filtering operations are commonly called smoothing in that they are low pass mechanisms which suppress rapid or spontaneous variations. Section 8.0 contains a detailed treatment of filtering and false alarm avoidance with emphasis on digital filtering.

Why are we interested in low pass filters or smoothers? When an IBV is operating properly, its performance measure signal will vary somewhat from sample to sample, this variation being quite small or characterized as "wandering." This is normal and should cause no problems in a design. If, however, the performance measure signal has a sudden extraneous point, a false alarm could well be displayed. Smoothing, then, is a mechanism for reducing false alarms. This smoothing can be accomplished prior to making a status decision or the status decisions can themselves be smoothed prior to status display. Predecision smoothing utilizes the wide body of recursive and nonrecursive digital filtering developed over the past several years. It is a numerical signal processing algorithm. Post-decision smoothing uses algorithms which exploit the filtering resulting from multiple event occurrences. It operates on the theory of recurrent events or runs. Section 8.0 addresses both types of smoothers.

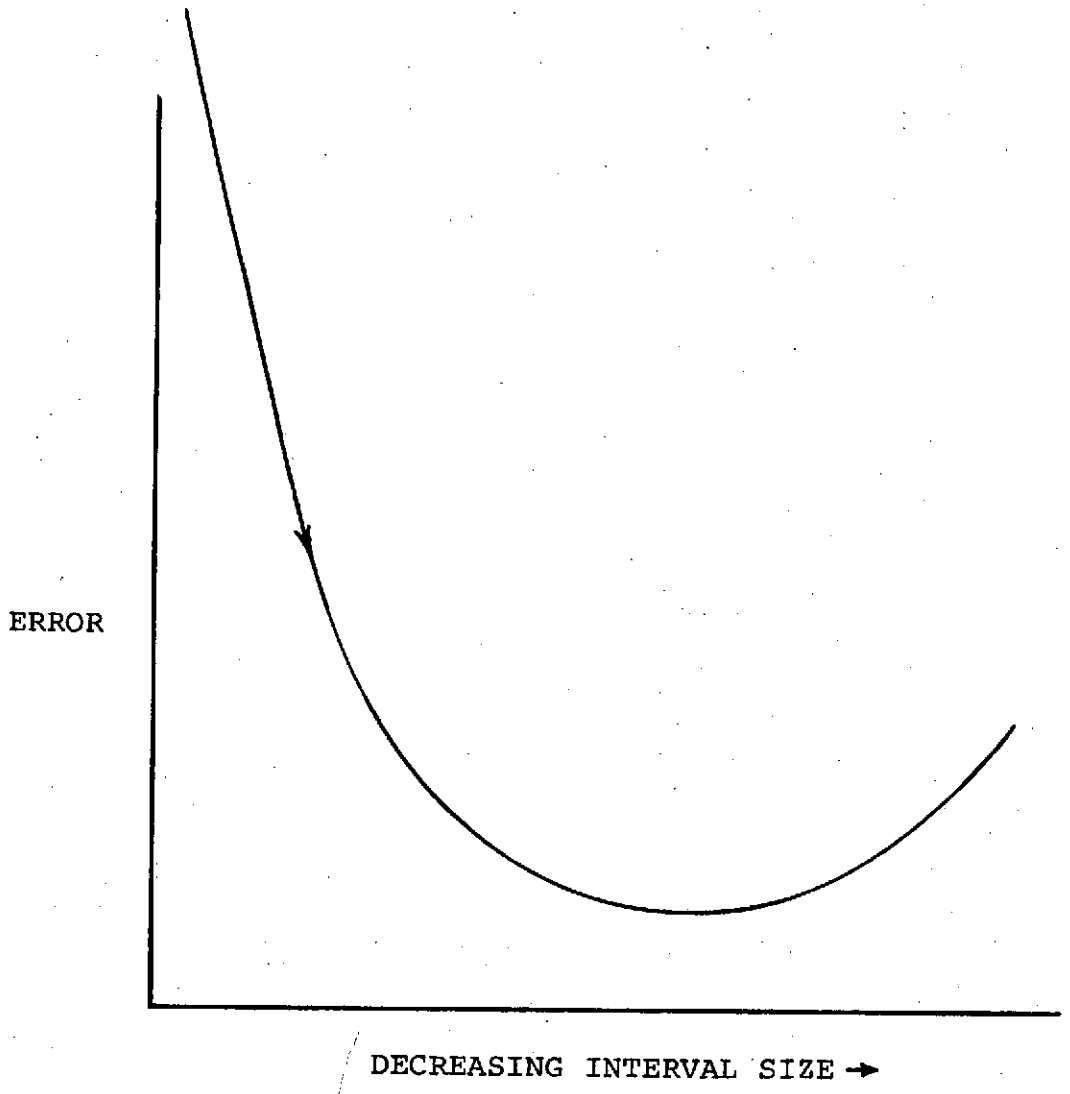



Figure 3.2-3. Solution Error vs. Numerical Integration Interval Size

If all one has to do is smooth data to remove false alarms, isn't the problem solved? By attempting to solve this problem, we have created a new one. It is true that in general, the more we smooth, the fewer will be the false alarms. Smoothing can be regarded as a damping process and, as is true with all dampers, delay is produced. Decreasing false alarms will inevitably decrease the likelihood of a timely notification of failure. This response, often expressed in delay to reach final value for a step input (sudden failure), is seldom of no consequence to the system user. A trade-off will typically be required between response time and false alarms. In the case of post decision filtering, the relationship is essentially the same, the degree of smoothing is traded against the time required to reach the recursion boundary for a step input.

This completes the verification process. We are now in a position to determine status of each IBV. Is this status indication the one which will be displayed to the crew? Not directly. The functional interrelationships among IBV's also means that the status thus developed is not independent of other IBVs' status. This relationship or dependence is the subject of the following section.

## 3.3 Status Resolution

Probably the easiest way to see the need for an intervening step between IBV status as developed above and the displayed information is by an example. Consider a series chain of IBV's. Assume the first IBV in the chain fails. The following IBV will no longer receive a proper input, and, will not likely provide an acceptable output. This same argument can be applied successively down the chain. What results is a host of failed indications from individual IBV's due to the failure of a single IBV.

The condition in the example cannot be allowed to remain since it violates one of the basic reasons for employing automated performance verification. A mechanism must be introduced which resolves the status indications of IBV's, selecting the "truly" failed IBV from several failure indications. One way of achieving this is store a system map or functional diagram which relates IBV flow and connections. Connectivity is the property we seek. When failed indications are indicated by IBV status, the status resolver will determine, based on IBV connectivity, which IBV has actually failed before any status information is displayed to the crew. In the above example, status resolution would pick the first IBV in the series chain as having failed, suppressing the failed indications from subsequent IBV's. Algorithms can be developed to take care of all conceivable block configurations (branch

in, branch out, series, parallel) except closed loops. Resolution within a closed loop will require interrupting the loop in the system.

# 4.0 PERFORMANCE VERIFICATION DESIGN APPROACH

The approach to design of automated performance verification is implied in the preceding sections. We shall summarize in this section. Flow of the design is depicted in Figure 4.0. It should be recognized that the design can, and will, fold back to preceding blocks as development on one portion of the design dictates changes in previous approaches. Appendix A to this volume contains an example of the design approach by applying the developed concepts to the design of automated performance verification for the Orbiter Hydraulics System.

The design begins by concisely identifying all IBV's such that there is no overlap of function or any ambiguity. This is the subject of Section 7.0 of this report and is shown as part of Block 1 in Figure 4.0. The second operation in this block has been touched on only briefly in this overview but is an important ingredient in the process. Recall that it is sometimes possible to determine the status of several functions using but a single instance of verification. It is this phenomenon which is being exploited here. The reader is referred to Section 7.0 for discussions of this subject. Suffice it to say that implementations of this sort should be pursued since they result in a decreased average processor burden. Block 3 in Figure 4.0 indicates the design of mechanisms which implement status determination of functions (here called elements) within an IBV. The mechanisms are called deduction schemes since they operate on logical relationships concluded from IBV status indications.

Returning to Figure 4.0, once IBV's have been identified which satisfy system degradation granularity, safety considerations and crew procedure cues, performance measures for each IBV must be developed. This is indicated by Block 2 in the figure. These measures are selected such that they best describe how well the IBV is performing its job.

Once performance measures are selected, limits must be set on these measures indicating the bounds of acceptable operation. These are the decision rules used by the mapper to determine IBV status from values of the performance measure. Block 4 indicates this step in the design.

Now that the parameters and design requirements have been established for each IBV performance verification, a design must be realized which meets these requirements. This is indicated in Block 5 as the design of status development — we are developing status for each IBV. Not explicitly shown in the figure is the smoothing operation for reducing false alarms. This operation is a must and has not been depicted due to the wide variability of its implementation.

Considerations in the design of the smoothing operation are false alarm rate and status response time.

Finally, Block 6 provides for the design integration and displaying of status. Status resolution design consists of identifying the principal system connectivity to the performance verification processor such that multiple failure indications can be resolved. Once resolution is effected, the true status indications, some of which are determined in Block 3, may be reported to a display device for crew information.

With the overview and design approach having been described, let us turn our attention to the details of specific verification components. These are discussed in the remaining four sections.

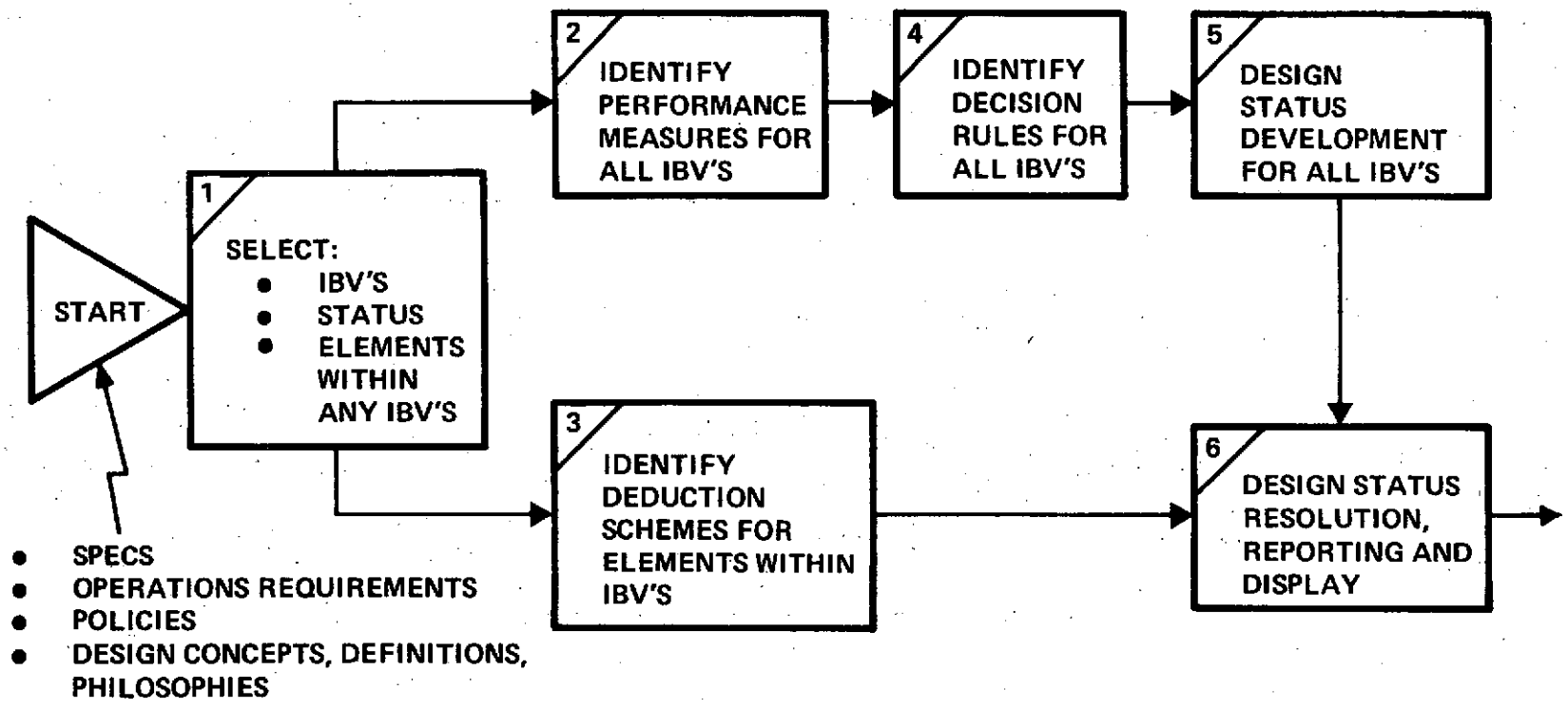



Figure 4.0. Automated Performance Verification Design Process

# 5.0 VERIFICATION TECHNIQUES AND MEASURES OF PERFORMANCE

Section 3.2 outlined the verification process and indicated the need for extracting measures of performance (MOP's) which are used for making IBV status decisions. Extracting these MOP's is the key to successful automated performance verification and is the subject of this section. Before proceeding with the discussion, it should be obvious that IBV's must be selected before MOP's can be identified. This is shown in the design process discussed in Section 4.0. How IBV's are selected is the subject of Sections 6.0 and 7.0. MOP extraction is being addressed first in the belief that it will aid in understanding of the overall verification problem. The reader should keep in mind that the order of these discussions is the reverse of the design process.

A measure of performance is selected in the belief that its value (numerical representation) best represents how well an IBV is performing. Also, this measure has the property that a value can be established which divides acceptable and unacceptable operation of the IBV. There is no restriction which forces the MOP to identically be a parameter of the IBV taken from a transducer. There is also no restriction that an IBV have a single MOP; it may be desirable to use several if a single measure will simply not be descriptive. The several MOP's, however, describe a good deal more than a collection of IBV parameters which are limit-checked. This distinction is important and should become clear in the following material.

The preceding relates one of the most fundamental notions regarding MOP's and warrants elaboration. First attempts at defining IBV performance measures usually involve the selection of numerous measured parameters. Seldom are all these parameters needed and equally seldom do they supply an adequate picture of IBV performance. They usually supply several small pictures which may or may not fit together to complete the IBV performance indication.

Having realized that "shotgun" parameters may represent an overkill, the next reaction would quite likely be to select a single, well chosen parameter as the MOP. Inherently, this is good practice since it unquestionably represents the cheapest way to go. There is, however, no guarantee that (a) there exists such a parameter or (b) that, if it could be identified, it would qualify as a performance measure. How can a parameter of an IBV not qualify as a performance measure? It surely indicates something of IBV behavior. What is wrong with this approach? Simply stated, it enters the back door of the problem. It is "solving" the problem in reverse. Let us address a possible forward approach.

Realizing that existing instrumentation is a genuine constraint, let us put this aside for the moment to consider the approach uninhibited by facts. Once IBV's are identified, the next step is to define precisely what each one does; what is its purpose? As often as not, this will be the signal, energy, material or what not that the next IBV(s) in the functional sequence are expecting to operate upon or "see." That is, the output of the IBV. The functional definition of the IBV, then, is usually the best start. Next arise a series of questions. Can this function/output be quantified? What about the dynamics of the function? Can it be measured while the IBV is performing its function within the principal system? Can acceptable/unacceptable boundaries be placed on the measure? Can the measure be derived from existing IBV information?

We shall take these questions one at a time. If the IBV function has been precisely defined, so too has the MOP. What remains is how to extract the MOP. There are three considerations here. The first is achieving a decision rule which is discussed in Section 5.1. The second is associated with achieving the MOP under IBV dynamic operation. This is the subject of Section 5.2. Finally, Section 5.3 addresses the problem of changing IBV operational characteristics under alternative operating modes.

## 5.1 The Influence of Decision Rules on MOP's

This is an appropriate place to stop and reflect about the use of an MOP. It will be used to make decisions regarding functional integrity of the IBV. This has been stated before, but this time we shall examine what it means from an implementation standpoint. If decisions are to be made using the value of the MOP, this implies that at any instant we must be able to state what that value should be. We cannot make a decision if there is no basis for the decision, if there is no fixed rule to go by. This factor is easier to violate than one may first imagine. Consider a communication channel for which the MOP would reasonably be the number of message errors. Now all that has to be done is count the errors and decide if the quantity exceeds a fixed number. This is fine but for one thing, if we don't know what the messages are going to be before hand, how do we know if they are in error. (It is interesting to consider the value of a communication channel for which one always knew what its contents would be.) In this case, the MOP (number of errors) indeed had a fixed decision rule and is, on the surface, a good communication measure. There is however, no reference from which to count errors. While a firm decision rule can be stated, there is no way to achieve the MOP using only the information contained in the channel. To

solve this problem, more information is added to the channel by way of error detecting codes. There is now a way to detect errors and realize the MOP. The error coding is deterministic.

This simple example points up two important considerations. First, it may not be possible to use existing information in the IBV to realize a MOP. Other information may have to be added so that it is possible to state what the MOP should be — and thus be able to make status decisions. The added information could be pilot tones, pseudorandom noise, time division multiplexed constants, etc. Decision rules can then be stated regarding MOP's for these types of information.

The second consideration pointed up by the example has to do with references against which to measure the MOP. In the example, we could not count errors until a deterministic feature was added. This feature in fact forms a reference on which the MOP can be based and will be necessary for every MOP. In this case we were able to state in absolute terms, that value above which the number of errors should not go, i.e., the decision rule or threshold value. Similarly, we can state that the value of a power supply voltage must remain within two definite boundaries. While this set of circumstances represents the easiest case to implement, it is not always possible. It may be that, if an MOP is to have a fixed decision rule, reference will have to be made to another part of the principal system. It may be that the only firm statement that can be made about an IBV performance involves an MOP which is referenced to another, similar IBV or to the input of the IBV. Such cases arise when one is unable to equate IBV output with its functional definition and at the same time state firmly what that output should be.

If an IBV's function is regulatory in nature, as in the case of the power supply cited above, it will usually be possible to identify an MOP with its output and be able to state the acceptable range of this output in an absolute sense. That is, be able to state a decision rule without regard to any other condition in the principal system. Many of the Environmental Control and Life Support functions aboard Orbiter will fall into this category. Many functions, unfortunately, do not operate in quite as neat a manner. If performance verification is to be practically realized, we cannot limit its application to the easy functions. We shall try harder.

Again, the key to IBV verification is an MOP which has a fixed decision rule. Alternative means of arriving at this goal have already been mentioned. Consider the class of

functions which are responsive in nature. Functions in a command segment are a good example. We cannot state in an absolute sense what the output of such a function should be. We can only state that their output should be the execution of the command which they received. That is, to achieve an MOP for such functions, we must state the output in terms of the input. One cannot say that a two degree right rudder is correct or incorrect until the input command is known. This is just one application for this class of functions. There are many more, all of which constitute varying degrees of sophistication on the same theme. Depending on principal system structure, such responsive functions will often perform transformations on the input command to effect its execution. Under these circumstances, these transformations, or their inverse, may have to be duplicated to achieve the MOP.

There is a third class of functions whose operation is so complicated that the only practical way an MOP can be described is to compare the results of two or more such functions. Digital computers, if rapid response time is necessary, almost always fall into this category.

The above discussion has been directed at methods of achieving fixed decision rules for an MOP. Three ways were identified.

- IBV output with respect to an absolute value.
- IBV output with respect to its input.
- The comparative outputs of two identical IBV's.

If one of these methods can be used, we are well along the way to achieving our purpose. For, to this point we have shown that a verification decision is realizable. What remains is the structuring of the MOP to the IBV performance. There are several recognized techniques for this and they are discussed in Appendix B. In the appendix, the techniques have been divided into the classes of achieving decision rules developed above. The necessity for status resolution (see Section 3.2) is also discussed, indicating whether conditional or unconditional status will likely result from application of a technique.

## 5.2 Implications of IBV Dynamics

Recall that the primary objective of performance verification is to effect this verification while the IBV is

performing its intended operation in the principal system. If this objective cannot be achieved, the IBV cannot be verified in real time. This section will discuss the implications of real-time verification on MOP selection and the techniques utilized. Once again, the problem is one of defining an MOP (or a minimum set of MOP's) which lend themselves to a fixed and meaningful decision rule.

Dynamical operation was not considered in Section 5.1 as it would have only served to clutter the issue. In the communication example given there, time was not considered in the message error MOP. The example of the power supply likewise did not consider time. The MOP's stated there are incomplete. In the communication example, the number of message errors must be stated per unit time (or equivalently, the number of errors contained in some constant number of messages). In the power supply example, depending on the mechanisms using the power, it may well be that an MOP which is concerned only with a constant voltage tolerance over all time is an over-simplification. More will be said about this problem in later paragraphs.

Dynamics affect performance verification from two almost opposite considerations. One of these considerations deals with the ability of performance verification to cope with, or operate effectively in spite of, anomalies which occur in MOP's resulting from IBV operation and the implementation environment. Data smoothing or false alarm avoidance is instituted to deal with this consideration. This is the subject of Section 8.0 and will not be pursued here except to mention that it is easy to lose sight of the smoothing objective. Smoothing is not a substitute for MOP definition or extraction; it is a mechanism for decreasing decision errors. These errors originate almost exclusively in the implementation, e.g., lack of data correlation due to sampled data implementations, and not from the IBV performance characteristics.

The second performance verification consideration resulting from dynamic IBV operation has to do with the ability to determine operational integrity. This is the subject of Section 5.2.1. Section 5.2.2 will pursue the same subject but with emphasis on sampled data implications.

### 5.2.1 Dynamics and MOP Selection

When an item is being performance tested, as for example during prelaunch checkout or acceptance testing, we have complete control of the item inputs, operation and output. To test a particular performance characteristic, the item is set up or configured in a prescribed manner and a prescribed,

simulated input is fed to the item. Its performance is then checked to verify compliance to specification. While this is a perfectly legitimate method of performance testing, it is rarely applicable to automated performance verification. Automated performance verification is attempting to determine operational integrity while the item under consideration is actually performing an operational task. The traditional stimulus-response (controlled cause-effect) methods seldom apply. Performance verification is a harder task. We must determine with an acceptable confidence whether an item is operative or not, without benefit of item control. This is not meant to imply that performance verification will achieve the same confidence in operational integrity as that of an acceptance test. It will seldom do so. The only claim that should be made is that the confidence is sufficient for the purpose.

Consider the power supply example discussed earlier. When a precision D.C. supply is bench tested, it first receives static tests under stepped loads to assure that it holds the specified voltage tolerance and ripple content over time and load range. The loads are exclusively resistive, i.e., no reactive components are used. The regulator dynamic response is next tested using these same loads. Of interest is the voltage recovery time and overshoot resulting from step load changes. These tests are designed almost exclusively to verify regulator, the most complex portion of any D.C. supply, operation. Another set of tests will also be performed to verify ripple and output voltage change due to input variations.

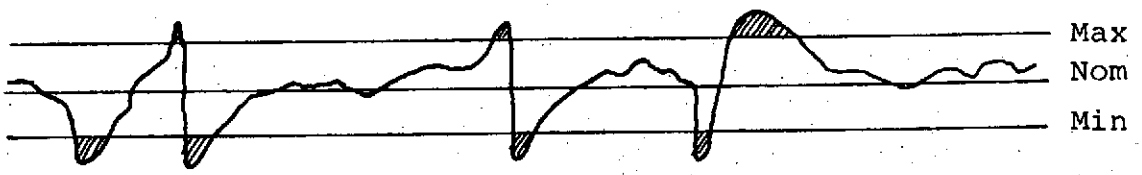
It is obvious that such tests cannot be performed on a supply while it is in use. In addition, the load on an operational supply is reactive which further complicates matters. It is equally obvious that a simple voltage range check, albeit smoothed, represents the other extreme of performance verification. Any implementation will usually be a compromise between these extremes, with the actual load power requirements being the deciding factor. If logic is being powered, dynamic response is quite important since the load can change abruptly at the logic clock rate. Affects of these changes are in practice reduced by distributed power filters. Also, some applications of a D.C. supply will, for all practical purposes, be insensitive to dynamic response. We shall, however, continue with this example since it is a convenient one and the results are applicable to any regulating-type IBV.

Interest in power supply dynamics centers about two parameters: overshoot and recovery. A regulator which has changed characteristics such that its closed loop gain has increased will be hypersensitive and overshoot upon step

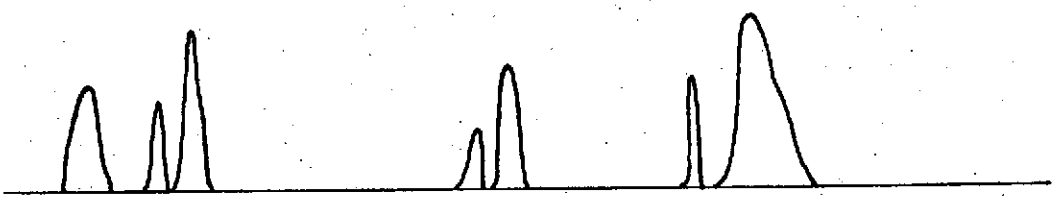
recovery. This will cause excessive voltage peaks and amost always, ringing. Recovery has to do with response time at step load. The concern is opposite of that above; it is a sluggish response that is of interest. A good measure of overshoot is the peak value reached by the output voltage. Recovery can be measured by some criterion which expresses the voltage-time product of excursions outside the voltage tolerance band. Root-Mean-Squared or Integral-Squared-Difference would both be suitable. The squaring operation is intended to give large excursions additional weight.

An alternative to the above is illustrated in Figure 5.2.1. Here, a single MOP is used which is defined as the RMS value of all out-of-tolerance excursions. Wave (c) in the figure shows the final MOP. Note that since averaging is being effected, there is, for most implementation, a practical limit on the time over which the average can be taken. Also, the averaging period is influenced by the behavior of the waveform being averaged. A waveform similar to that in Figure 5.2.1(c) can be achieved by an integrate-and-dump device which is easier to implement than the root-mean device. The decision rule for this MOP would be stated in terms of the averaging period RMS out-of-tolerance value, rather than a voltage threshold corresponding to the maximum and minimum limits. Figure 5.2.1(d) illustrates the values which would be observed by a direct sampling of the original waveform at sample times corresponding to the ends of the averaging periods. Note the lack of sensitivity to the voltage excursions. This type of sampling would be effective in detecting a change in regulator reference voltage but will likely not detect fluctuations which could cause, for example, errors in logic. Smoothing the waveform in (d) would only aggravate the situation. Again, smoothing is not intended to replace MOP techniques.

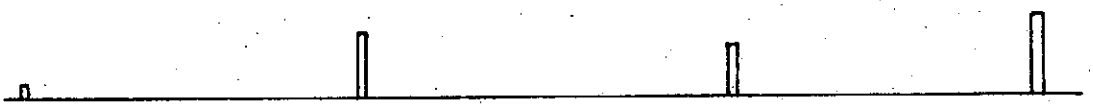
The example is intended to demonstrate the development and extraction of an MOP. It is likely that an IBV will be (and should be) larger and more complex than a single power supply. Whether the MOP should actually be applied to a power supply depends on a large number of factors. The cost of the MOP extraction is not trivial and should obviously be justified by the need. Distributed capacitance throughout the D.C. distribution goes a long way toward smoothing voltage peaks but energy is still required to accommodate a sudden increase in load. The value of maximum and minimum voltage may well be nominal $\pm 10\%$. For much of today's integrated circuit technology, this amounts to $\pm \frac{1}{2}$-volt. This is a small value when one considers the susceptability of regulator sense lines to noise, hum, cross talk, etc.



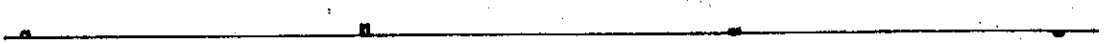

(a) Original Waveform



(b) Squared Difference



(c) RMS At End Of Averaging Periods



(d) Sample of Waveform in (a) at Averaging Periods

Figure 5.2.1. Development of a Power Supply MOP

The selected MOP comes a lot closer to measuring the dynamic function of a regulated power supply than simple voltage measurements. This is especially true where voltage sampling is concerned, which is the topic of the following section.

The reader should note the broad selection of verification techniques contained in Appendix B. These techniques are intended to describe possible ways of achieving MOP's and are far from complete. The combinations and possibilities are limited only by the designer's imagination.

### 5.2.2 Implications of Sampled Data

We shall continue with the power supply example in Section 5.2.1 above. The implications of sampling the voltage waveform have already been discussed. If the application is not sensitive to voltage excursions but only the "average" voltage, then voltage sampling is probably adequate. This also implies that a voltage level MOP would also be adequate. What if the MOP in Section 5.2.1 must be used? Where does sampling fit in then?

To begin with, it will likely be impractical to sample at a rate frequent enough to develop the waveforms in Figure 5.2.1. This implies local (as opposed to CPU) processing of the MOP; quite likely in analog form. This will be the rule rather than the exception for any IBV where the shape and/or character of its output is important. The CPU would then sample the RMS values in Figure 5.2.1(c) and process these values just as it would have processed direct voltage samples. In fact, in the example, the averaging period would correspond to the sampling period.

The local analog processing of MOP's can pose a serious problem to a system that has not made provisions for such an eventuality. Required, will be additional sample time slots for the MOP's (since, in the example, the power supply voltage will still be of value for other purposes), physical space for the circuitry, additional power and thermal control. Unless an adaptive sampling protocol can be implemented, there is likely no compromise between the direct voltage sampling and the local MOP processing. With direct parameter sampling, there is no way to achieve information about the IBV output which occurs, in practice, at frequencies above 1/8 the sample rate. There will certainly be valid MOP's that can be formed on this basis and there will, with equal certainty, be MOP's which demand local processing.

## 5.3 Implications of IBV Operating Mode Changes

To simplify relationships, all the preceding discussions addressed verification as though IBV's would remain in a single operating mode. This is obviously rarely true. IBV's will change modes and, if nothing else, will be turned on and off. It is not possible to verify performance of a powered down IBV. Thus, the performance verification system must be both notified of this condition and be able to accommodate it so that the IBV will not be declared inoperative.

Accommodating operating mode changes is highly dependent upon the particular IBV and the individual modes. Some mode changes will allow the continued use of previous MOP's, reguiring only a change in their decision thresholds. Others will require different MOP's and still others may simply delete an MOP. The important result of all this is that the modes must each be defined, MOP's (along with decision thresholds) must be defined for each and the verification system must be able to adapt automatically upon mode change signals from the IBV.

## 6.0 CRITERIA FOR SELECTING FUNCTIONS TO BE VERIFIED

The purpose of this section is to identify criteria for selecting Orbiter Subsystem functions which are most amenable to real-time automated performance verification. The criteria are intended to be used in conjunction with the general techniques described in Section 7.0. While the latter generally identifies items to be verified, the criteria in this section are used to determine how the verification is best accomplished, or indeed, if it should even be accomplished on an automated basis. The performance verification techniques identified in Section 5.0 were used as the basis for determining the criteria.

### 6.1 Factors Affecting Selection

When identifying criteria for selection of functions to be verified, there are four factors which must be considered:

a. The operational constraints.

b. Attendant verification programming complexity and storage requirements.

c. Implementation.

d. Mission time during which verification is accomplished.

Each factor represents a different interest in a design realization and should be considered independently. By doing so, the criteria can be used not only for selecting functions for verification but, in a much more positive sense, to guide the separate design interests in achieving a final design which is more amenable to verification. Note that attendant verification programming complexity and storage requirements (factor b above) is the only factor which is associated directly with the function itself. The ramaining factors are associated with how the function is to be used.

The selection criteria appear in Table 6.1 where they are grouped by the identified factors. As might be expected, some of the criteria appear under more than one factor. Each factor, together with its associated criteria, is discussed in Sections 6.2 through 6.5.

### 6.2 Function Selection by Operation

This set of criteria deals with the ability of the crew to:

a. Observe failure symptoms.

Table 6.1. Function Selection Criteria

| FACTOR | CRITERIA |
|---|---|
| A. Operations | Select for automated verification, functions:<br>1. Whose failure is not immediately obvious to the crew.<br>2. Whose failure may be immediately obvious but the consequences of such a failure will represent a hazard to crew.<br>3. Once failed, will require action on the part of the crew for recovery/reconfiguration.<br>4. Requiring automatic reconfiguration. |
| B. Programming complexity and storage requirements | Select for automated verification, functions:<br>1. Which do not operate in several modes.<br>2. Whose operation can be judged against a constant value rather than against operation of another function (relative) or against a calculated variable dependent on time, velocity, mission phase, etc.<br>3. Other than command and control. |
| C. Implementation | Select for automated verification:<br>1. Functions which are relatively large.<br>2. Functions whose outputs are accessible.<br>3. Redundant functions whose outputs are discernable.<br>4. Redundant functions which are identical.<br>5. Functions not within a closed loop. |
| D. Frequency of Verification | Defer to post-flight verification, functions that:<br>1. Are very reliable.<br>2. Are incidental to safety and/or mission.<br>3. Can not be reasonably verified real time (due to complexity, confidence, accuracy).<br>4. The failure of which will result in inability of crew to recover (commitment).<br>5. The failure of which will not alter the mission or safety when the function is used during initial portions of flight only. |

b. Readily associate the symptoms with a specific function, and

c. Ability/necessity of the crew to recover from the failure. Note that recovery does not necessarily imply replacement of a failed function in kind. Rather, it generally implies prevention of disaster following failure.

If the crew can readily observe the failure symptoms and uniquely associate these symptoms with a function, there is little need to incorporate automated verification so long as failure recovery time is not critical. If recovery time is critical then automatic recovery, and obviously automated verification, will be required. If the crew cannot immediately associate symptoms with a unique function, recovery time will be protracted while the failure is being isolated.

As there are few functions the failure of which will produce only incidental deleterious effects on safety/mission, it is safe to assume that the crew requires knowledge of all failures. Why? The answer resides in the crew's ability to do something about the degraded state. The crew will presumably act in a different manner when the Orbiter is in a degraded state -- defer maneuvering, scrub experiments, switch in a redundant function. If a state exists which commits the crew to ultimate disaster, i.e., there is nothing they can do to recover, it is questionable whether they need even be informed on an automated basis. If an emergency procedure exists, the crew needs to know the system state to execute the procedure. If one does not, the crew is no worse off than if they had no information.

## 6.3 Function Selection By Programming Complexity And Storage Requirements

This set of criteria deals with properties of functions which translate into automated verification programming complexity and storage requirements. Consideration has been given to storage requirements for both program and data.

Verifying functions that operate in several different modes inevitably requires more storage and some added programming complexity. Such functions require mulitple verification routines and, typically, mode switching algorithms. Moreover, additional data must be input and stored to account for mode changes.

It is easier to verify a function whose operation can be judged against a constant value rather than a variable which must be calculated or determined based on several other

function conditions. The latter requires added programming sophistication and storage.

Automatic, real-time verification of command and/or control is always a challenge. Command functions inevitably involve, from the standpoint of the verification equipment, random operations with only sophisticated correlation functions among events. Stated another way, command information as a class usually originates from the crew rather than from within independent operation functions. As such, there is seldom any basis for automatically verifying operation except by duplicating the function and comparing results. Command functions can be verified much more easily on a manual basis by feedback information which indicates to the crew the command was actually executed.

Control functions typically suffer from the same uncertainties characteristic of command. In addition, control usually operates within closed loops. Automatic verification within such a loop is quite difficult.

## 6.4 Function Selection By Implementation

This set of criteria deals with the mechanization of the functions and its effects on verification. Due to inevitable error possibility, and due to the fact that it is often as difficult to verify a small (from a complexity standpoint) function as a large one, it is better to establish verification for functions that are as large as possible. For the same system, this policy reduces the number of verification decisions which must be made and consequently decreases overall verification error. The upper bound on the size of verified functions is obviously the resolution of the functional status demanded. Note that this concept can also be extended to include reliability of the verified functions. Signals must be extracted from the function being verified to be used for verification purposes. This extraction involves hardware — sensors, isolators, multiplexers, etc. The reliability of this hardware should be much greater than the reliability of the verified function and the easiest way to achieve this relationship is to verify large and/or relatively unreliable functions. This last point is based on an argument advanced earlier that often the instrumentation requirements for large and small functions are not appreciably different.

The principal source of information used to judge the operational integrity of a function is its output. For this output is the function's reason for being. It should go without saying that the output must then be accessible to the verification system. There are numerous reasons why the output may not be accessible, some of which are:

a. Multiplexer and sensors were not incorporated into the design.

b. Output is electrically/mechanically "buried" within the system.

c. Due to electrical/mechanical connections (usually resulting from redundancy applications), the output cannot be discerned from that of another item being verified.

d. Output is so complicated or consists of so many variables that it is not practical to be considered accessible.

As indicated in (c) above, the ability to discern outputs of redundant functions is important. If it is required to know which function in a redundant set has failed, the outputs must be discernable. "Hardwired" redundancy is perhaps the largest single offender of this principle as the outputs form a single electrical node.

To carry the above notion one step further, it is usually easier to verify redundant functions which are identical (as opposed to those which provide lesser capability in the event of prime function failure). Under these circumstances one seldom cares which function is failed so long as he knows how many have failed. The outputs then do not have to be discernible. In addition, a single algorithm can be used for verification, thus reducing programming and storage requirements.

Section 6.3 above indicated the problems involved when attempting to verify functions within a loop.

## 6.5 Function Selection By Frequency Of Verification

This set of criteria deals with how often the function can be verified. Not all functions need be verified on a continuous basis.* Some may be used only periodically, others for one brief period during the mission and still others may be reliable enough and/or noncritical enough to warrant lengthy response times and risk of not detecting a failure as soon as it occurs. The ultimate in verification frequency is a one-time thorough verification at the beginning of the mission. The implications of this approach

*As used here, continuous, does not necessarily imply analog monitoring. Rather, it implies a verification rate (sampled data techniques) which is greater than the natural response of the verified function.

are obvious: there would be no need for onboard fault detection. Unfortunately, nature is seldom so cooperative. In the final analysis, the Orbiter will employ extensive, high-confidence ground checkout which will verify the bulk of performance and be the only verification of some devices. A lower confidence, real-time verification will be used for most, but not all, devices. These points will be pursued below under a discussion of the criteria. First, however, some additional philosophy must be examined.

As with most large systems, the Orbiter has functions which are dormant during part of the mission. The question arises, "when is a failure in these functions?" If a dormant function fails, the system, i.e., the Orbiter, certainly has not failed since this function was not even being used. At the time the function is finally required, however, the system at large will not perform as required. Assuming the unused function can even be verified, is it worth while determining its status before it is needed? During ground checkout the answer is obviously yes. Whether in-flight verification is required can be tested in each case by whether knowledge of a future event will affect current activity on the part of the crew. When viewed in the cold light of day, knowledge of this future event will likely not alter current mission phase activity. If, while on-orbit it is discovered that the landing gear will not operate, there seems to be little use in aborting the mission. The foregoing is not meant to imply, however, that point verifications (as a minimum) should not be made just prior to changing mission phases. In the example, it may be prudent to attempt a gear check prior to de-orbit as this information may alter re-entry characteristics and timing. As a general policy, all equipment to be used in a subsequent mission phase should be verified before commitment to that phase.

Returning to frequency of verification criteria, if a function is very reliable (which many times equates to a "small" function), it is questionable whether automated verification is necessary unless the function is switched redundant. If the chances are very remote that a function will fail during a mission, ground checkout should be sufficient as real-time verification can only add to confusion when a decision error occurs. The same argument can be advanced for incidental functions.

It will simply not be practical to verify some functions on a real-time basis. This verification will have to be relegated to ground checkout. If necessary, such functions can, however, often be verified in-flight at various points during the mission.

It is possible that, especially during initial operational flights, occurrence of some functional failures will leave the crew incapable of recovering. Such events are

rare and are usually associated with a peculiar set of mission executions. In this light, it is questionable whether automated verification will yield cost effective, meaningful information.

Finally, when a function is used during initial portions of the flight only, (prior to injection) there should be little need for real-time verification. Ground checkout should provide adequate confidence in these functions and once their usefulness has passed, so too should concern for their operational integrity.

# 7.0 TECHNIQUES FOR SELECTING ITEMS TO BE VERIFIED

Section 6.0 identified criteria for selecting functions or items to be verified. This section extends those results by developing a systematic method for approaching the problem. The method initially identifies candidate IBV's without regard to implementation considerations. Once IBV candidates are identified, implementation considerations enter into scrubbing the candidates to arrive at the final selection.

Of the design factors considered in Section 6.0, all are implementation-oriented except one; operations. Operational considerations mark the initial thrust of our development. The criteria identified in Table 6.1 for this design factor have recovery requirements as the common denominator. What is the principal reason for verifying the operational integrity of functions? Simply stated, this information is necessary to initiate recovery actions. It follows that:

a. The information must be at the same level as the recovery action.

b. For Orbiter, recovery represents the level at which the crew can do something about a failure.

The crew can recover from failures in numerous ways. They can modify mission objectives, modify maneuver executions or abort. These are all procedural methods of recovering or being able to survive, albeit at reduced capability. Functional redundancy presents another recovery option. It is a designed-in recovery from functional failure which is effected by system reconfiguration. Redundancy, then, represents a major, tangible recovery mechanism and is one of the most important factors in identifying IBV's. Section 7.1 presents an outline procedure for identifying IBV's based primarily on principal system redundancy structures. Once IBV's have been identified, it will be necessary to examine how each is to be verified. It may not be necessary to actually verify each IBV by the methods contained in Section 5.0. There are some additional techniques which can be used to determine IBV status without actually verifying it directly. These are known as status deduction techniques and are discussed in Section 7.2.

## 7.1 IBV Identification

This section presents a systematic approach to identifying IBV's. All reasonable candidates are first

identified and these are then modified as necessary when implementation is considered.

### 7.1.1 Step 1 — Preparation of Functional Flow Diagram

An accurate functional flow diagram of the principal system is an imperative beginning. The detail of the diagram should be at the level of recovery interest. All redundancy at this level must be shown and redundancy interfaces will likely require detail to the piece part level. Each case of redundancy should be supplemented by a redundancy definition document which details exactly how the redundancy is achieved and recovery is implemented.

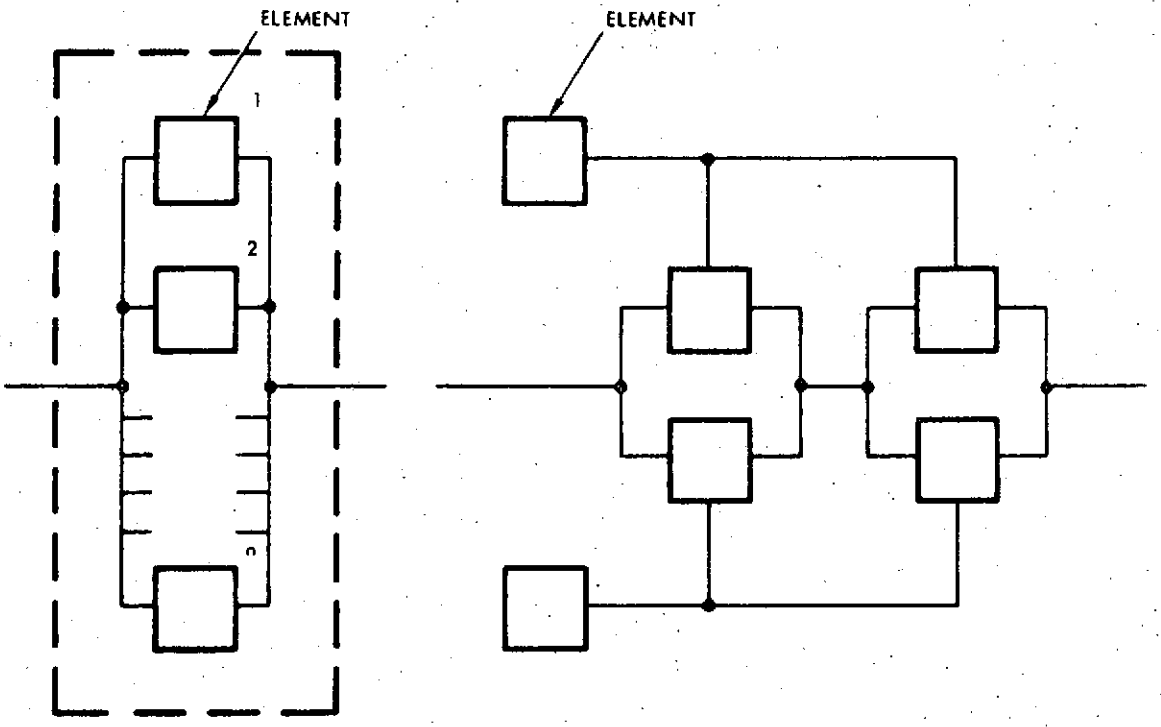
### 7.1.2 Step 2 — Identify All Redundancy On The Diagram

Indicate, with the use of constraining lines or boxes, the principal system recovery partition. This partition will likely be different from design partitions and must be done with care. Of interest are the individual elements which are involved in recovery procedures. Such a diagram can be more closely identified with the task analyses used in deriving emergency procedures.

The important information to be gained from this step is the identity of each redundant (and nonredundant) recovery entity and their relationships to other elements. Of the redundant elements, two distinct patterns will appear. One will be the classic parallel redundancy wherein the redundant elements all have the same predecessors and followers from a functional flow sense. That is, there will be a functional node at both the input and output of the redundant network where all the network elements are tied. The second pattern which will appear is more complex. This pattern results from relationships among elements which preclude them from being configured as parallel redundancy. Such patterns often result from functions which are used for multiple applications in the principal system. For convenience, we shall call patterns of the first type redundant sets and those of the second, redundant groups. (See Figure 7.1.2).

### 7.1.3 Step 3 — Identify IBV Candidates

Essentially, each element identified above is a candidate IBV. Depending on the redundancy definitions and element relationships, some of these candidates can be deleted or modified. The first area which should be examined are series strings of nonredundant elements. Unless there are compelling procedural reasons to the contrary, all such series elements can be lumped into a single IBV. Verification information on a single element in such a series




(a) Redundant Set (b) Redundant Group

Figure 7.1.2. Examples of Sets and Groups

chain provides no more recovery information than information relative to the entire chain.

The redundant sets are the next area of interest. It is possible that each element of such a set need not be an IBV. If the elements all perform the function with equal capability, i.e., they are all identical, and if the crew does not need to know, specifically, which element has failed, it may simply be necessary to determine the number of operative elements (as opposed to the status of each element). This action would make the entire set an IBV.

Finally, the redundant groups should be examined. These typically pose the most difficult verification problems and it is worth additional effort to either simplify the group by redesign or eliminate IBV's with some reduction in crew recovery information.

### 7.1.4 Step 4 —IBV Implementation Evaluation

The implementation considerations of Section 6.0 are now applied to the candidate IBV's identified thus far. These considerations are expressed by the criteria associated with Factors B, C and D in Table 6.1. This step identifies whether the IBV's should be verified on a real-time basis or a deferred-time basis. MOP's will have to be identified in either case, the major consideration being whether the IBV verification needs to be included in real-time CPU verification routines or not. For example, most redundant, off-line elements need not (and indeed cannot) be verified on a real-time basis.

Deferred-time verification consists of two approaches: in-flight verification and pre-flight verification. The former is executed at scheduled points during the mission and will have to be accommodated in the CPU. It will also require provisions for MOP extraction and data acquisition slots. The latter are the responsibility of Ground Checkout and no Orbiter provisions should be necessary. IBV's which fall into this category are those which are very reliable or which are incidental to safety and/or mission.

This completes the identification of IBV's. We now turn to some schemes which may be used to reduce the number of resulting MOP's.

## 7.2 Status Deduction Techniques

Is it essential that each IBV have a minimum of one MOP? Development of MOP's can be costly and one is motivated to delete as many as possible without jeopardizing recovery

information. Fortunately, there are two ways of reducing the number of MOP's. One way is termed logical deduction and the other, iterative deduction. They are each discussed below. Using these techniques, it is possible to obtain the status of IBV's for which no MOP's are extracted. This is done by capitalizing on logical relationships among the IBV's, thus the name, deduction.

Logical deduction consists of establishing algorithmic relationships between the status indications of several related IBV's. Each implementation represents a special case and will require individual treatment. A general example is, therefore, hard to come by, but the principle operates as follows: if the status of A and B are known, one can conclusively say what the status of C is without ever having checked C. Candidates for logical deduction are most likely to be found in fan-in or fan-out functional arrangements.

Iterative deduction, as the name implies, consists of replicating system configuration until the failed IBV is identified. The technique only works for switched redundancy where off-line functions can be readily switched on-line and where sufficient time exists to allow the replications to be effected.

The technique works as follows. Consider a chain of switched redundant sets where each on-line element is an IBV but only one MOP (or set of MOP's) is defined at the end of the chain. When an on-line element fails, this condition will be reflected in the end-of-chain MOP. This MOP can obviously only reflect the state of the entire on-line chain. Since each on-line element is an IBV, we begin switching off-line elements into the on-line chain until the MOP indicates the failed condition has been corrected. At this point we have also identified the failed IBV.

If the functions performed by the redundant chain are not critical, this technique is quite practical and results in potentially large savings. The switching can be initiated either manually or automatically. Note that success of the technique depends on off-line elements which are operative. To assure this, their status should be verified occasionally. If the off-line elements are kept de-energized, their warm-up time must be considered in the time estimates for this technique.

## 8.0 SMOOTHING TECHNIQUES

The objective of this section is to survey several smoothing techniques which are considered to be the most likely candidates for implementation in automated performance verification. Each technique is designed to be implemented on a digital computer.

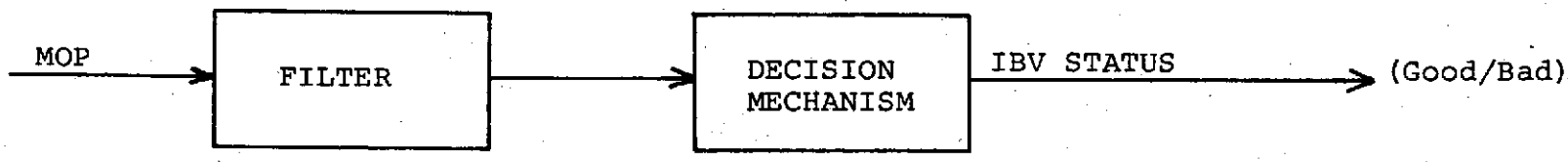
Recall from Section 5.0 that the purpose of smoothing is to reduce decision error and in particular, false alarms. To achieve this end, the process must prevent decisions from being made on information anomalies, thus the name smoothing. It will be shown in the discussions that, by introducing smoothing to reduce false alarms, the likelihood of a miss is increased. This is a characteristic of all smoothing operations.

The smoothing techniques surveyed can be grouped into two distinct approaches according to their operating principles. The first approach we shall call predecision filters and the second, post-decision filters. The operation of the two approaches is depicted in Figure 8.0 from which the reason for their names should be obvious. The predecision filters discussed are recursive, digital, low-pass filters. These are addressed in Section 8.1.

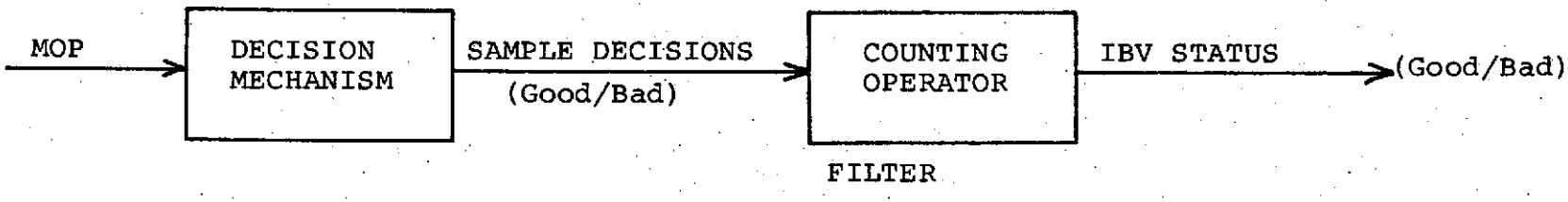
The post-decision filter, addressed in Section 8.2, requires a bit more explanation than its counterpart above. First, as evidenced in Figure 8.0, this class of filter operates on binary information. It effects smoothing by tallying decisions in a given run and effectively voting on the results. The process is altered by changing the voting rules and the size of the run considered. For example, one method of achieving this type filter is to require N consecutive Bad decisions at its input before IBV status is declared Bad. This in effect requires a unanimous vote of Bad decisions before the filter declares a Bad decision. Another method of designing the filter is on a plurality basis, i.e., N bad decisions required out of any M total decisions.

### 8.1 Predecision Filtering

There are numerous ways of achieving this class filter since it performs in the classic role of a low-pass filter. We shall restrict our interest to a class of filters known as recursive smoothers which are by far the most applicable to automated performance verification use. These filters exhibit behavior quite similar to their analog counterparts. We shall describe the first order smoother in some detail in Section 8.1.1, with higher order smoothers being covered collectively in Section 8.1.2.




(a) Predecision Filter




(b) Post-decision Filter

Figure 8.0. Operation of Two Approaches to Smoothing

## 8.1.1 First Order (Exponential) Smoother

Figure 8.1.1-1 shows the mechanization of this smoother. Here, T is the time between samples, X(KT) is the value of the Kth sample input, Y(KT) is the value of the smoother output at the Kth sample, KT is the Kth sample time and $\alpha$ is the smoothing constant. The follwoing several steps illustrate the recursion.

$$Y(KT) = (1-\alpha)Y(KT-T) + \alpha\, X(KT)$$

$$Y(KT+T) = (1-\alpha)Y(KT) + \alpha\, X(KT+T)$$

$$Y(KT+2T) = (1-\alpha)Y(KT+T) + \alpha\, X(KT+2T)$$

$$\vdots$$

The general recursion relation is,

$$Y(KT) = \alpha\, X(KT) + (1-\alpha)Y\left[(K-1)T\right] \quad ,$$

$$0 \leq \alpha \leq 1$$

As may be apparent from this relation, the smoother may be viewed as a sampled data equivalent of a single pole filter. This technique applies geometrically related weights to successive time samples to achieve the estimate very similar to the impulse response of a filter. Intuitively, the net effect is to modify the influence which each immediate sample has with respect to the output. Thus for a value of $\alpha = 0.5$, the history of the filter and the present value of input receive equal weighting. As $\alpha$ is reduced beyond 0.5 more emphasis is placed on the past history of the input. The weights applied to the individual samples are,

$$\alpha(1-\alpha)^m$$

where m represents the sample taken m periods ago. Large values of $\alpha$ will yield a responsive smoother and small values, a more stable smoother.

We should like to be able to rate the smoothing performance of the filter for purposes of selecting $\alpha$ and for purposes of comparing filter alternatives. Expressed in such a measure is the false alarm immunity of the smoother. The most general method of achieving this measure is to compare the variance of the smoother output to the variance of its

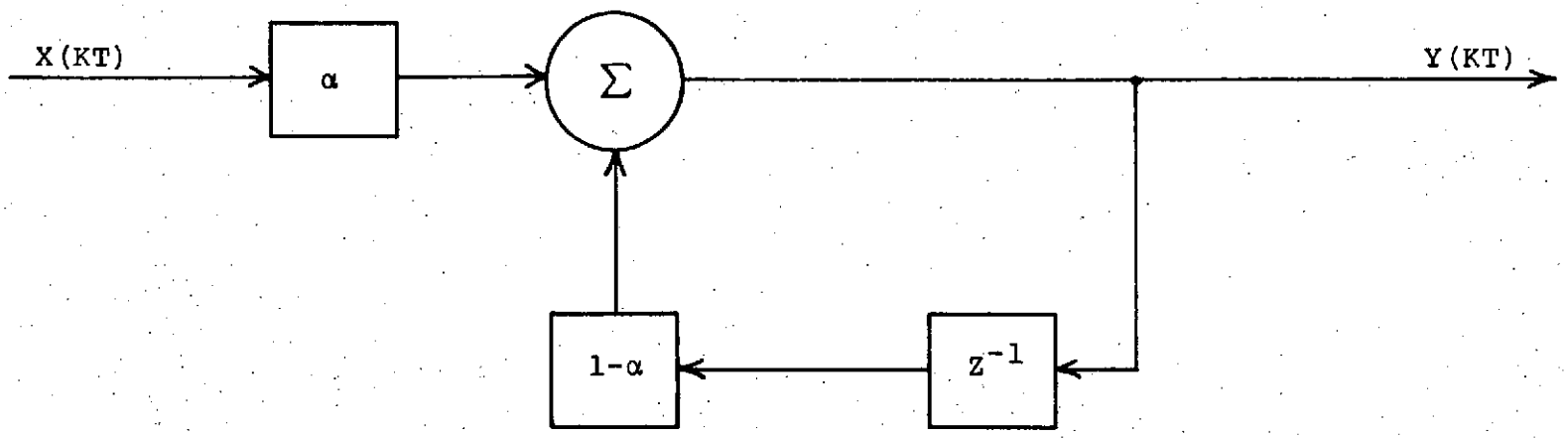



Figure 8.1.1-1. Mechanization Of A First Order Recursive Smoother

input. For, the purpose of the smoother is nothing more than to reduce input variance. The output, Y(KT), variance is expressed

$$\sigma_e^2 = \frac{\alpha}{2-\alpha} \sigma_s^2$$

where $\sigma_e^2$ is the variance of the filter estimate, i.e., its output, and $\sigma_s^2$ is the variance of the input sample. The ratio of these two variances is the measure we seek and is plotted in Figure 8.1.1-2. It should be noted that the expression of output variance above makes two assumptions.

- Values from the random process causing the variance will be uncorrelated from sample to sample.
- The time between samples, T, is constant.

If either of these assumptions does not hold, the output variance will be greater than that indicated by the operation.

A second measure of smoother performance is its step response. This expresses its immunity to misses. The step response is expressed as the number of sample periods required to reach 90% of the final value. This relation is also plotted in Figure 8.1.1-2. It should be noted that the step response represents the worst case response of the smoother. All other inputs will exhibit a response less than the step response. A smoother can be designed graphically from Figure 8.1.1-2 by trading off false alarm immunity and miss immunity in the selection of $\alpha$.

In terms of complexity, the first order smoother is simple to implement. It requires but a single data storage location, the value of Y(KT), and a simple arithmetic program.

### 8.1.2 Higher Order Recursive Smoothers

Just as higher order analog filters (increased number of poles) are used to achieve equivalent response with a smaller bandpass, higher order smoothing may be used to accomplish the same end. In general, n-order smoothing is realizable but interest here will be limited to no greater than third order.

Second order smoothing essentially imbeds the first order process in an identical but overlying process. Operationally, second order smoothing determines the first order

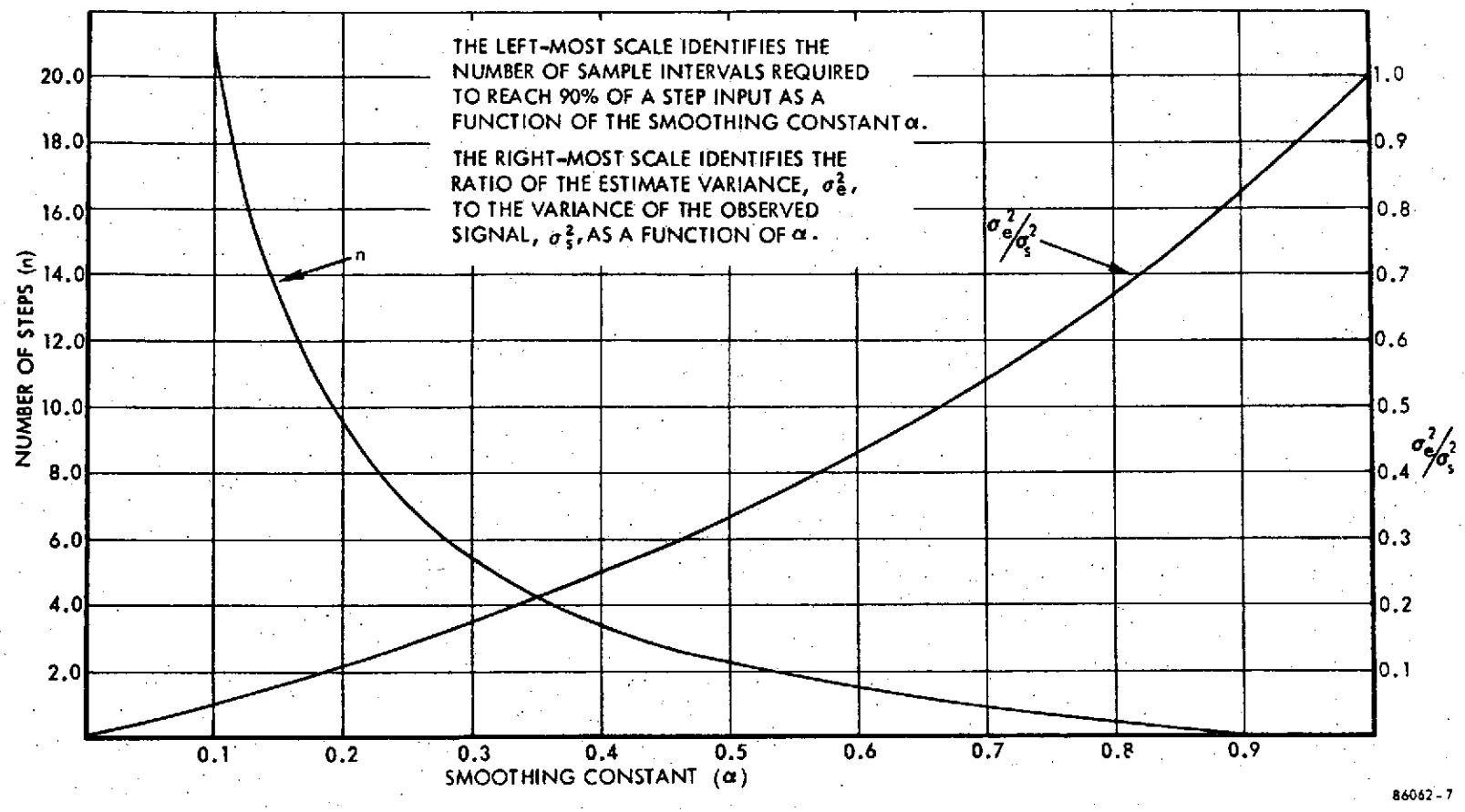



Figure 8.1.1-2. Performance Characteristics of a First Order (Exponential) Smoother

estimate and uses this estimate (as opposed to the observed value) to update the second order estimate. Then,

$$Y_2(KT) = \alpha Y(KT) + (1-\alpha)\left[Y_2(K-1)T\right]$$

where $Y_2(KT)$ is the second order estimate at time KT (or period K) and Y(KT) is the first order estimate. This technique will require two locations in memory since both $Y_2(KT)$ and Y(KT) must be preserved for the next period. (Recall that $\alpha$ must either be stored explicitly or entered as a constant in the program.)

Third order smoothing continues the trend set by second order and operates on the second order estimate,

$$Y_3(KT) = \alpha Y_2(KT) + (1-\alpha)Y_3\left[(K-1)T\right]$$

As might be expected, third order smoothing requires three memory locations.

A comparison should be made between the three smoothing techniques with regard to response. The basis for comparison is response to a step input. If $\alpha_1$ is the smoothing constant for first order smoothing, $\alpha_2$ second order smoothing etc., an equivalency is shown in Figure 8.1.2. From this figure, it can be seen that, for an equivalent response to a step input, a second order smoother would require an $\alpha$ of 0.37 and a first order smoother an $\alpha$ of 0.6. These results can be related to the response plot of the first order smoother in Figure 8.1.1-2. For $\alpha = 0.6$, a first order smoother would reach 90% of its final value in two steps. A second order smoother would achieve the same result with an $\alpha = 0.37$. While the variance curve for second order smoothing is not shown, it will exhibit the same trend as that shown for first order. It is obvious that a reduction in estimate variance (with respect to the variance of the observed signal) is achieved.

Some further characteristics are summarized in Table 8.1.2. The impulse response is valuable for determining rejection of spikes and providing comparisons with comparable analog filters. The transfer function will afford Z-plane analysis.

## 8.2 Post-decision Filtering

This filtering operates by combining successive status decisions to generate an alarm state. Such combining

Table 8.1.2. Some Characteristics Of The First Three Orders Of Recursive Smoothing

| ORDER OF SMOOTHER | IMPULSE RESPONSE | TRANSFER FUNCTION | REQ'D NO. OF STORED DATA WORDS |
|---|---|---|---|
| First | $\alpha(1-\alpha)^t$ | $\frac{\alpha}{1-(1-\alpha)z}$ | 1 |
| Second | $\alpha^2(t+1)(1-\alpha)^t$ | $\frac{\alpha^2}{[1-(1-\alpha)z]^2}$ | 2 |
| Third | $\alpha^3\frac{(t+1)(t+2)}{2}(1-\alpha)^t$ | $\frac{\alpha^3}{[1-(1-\alpha)z]^3}$ | 3 |

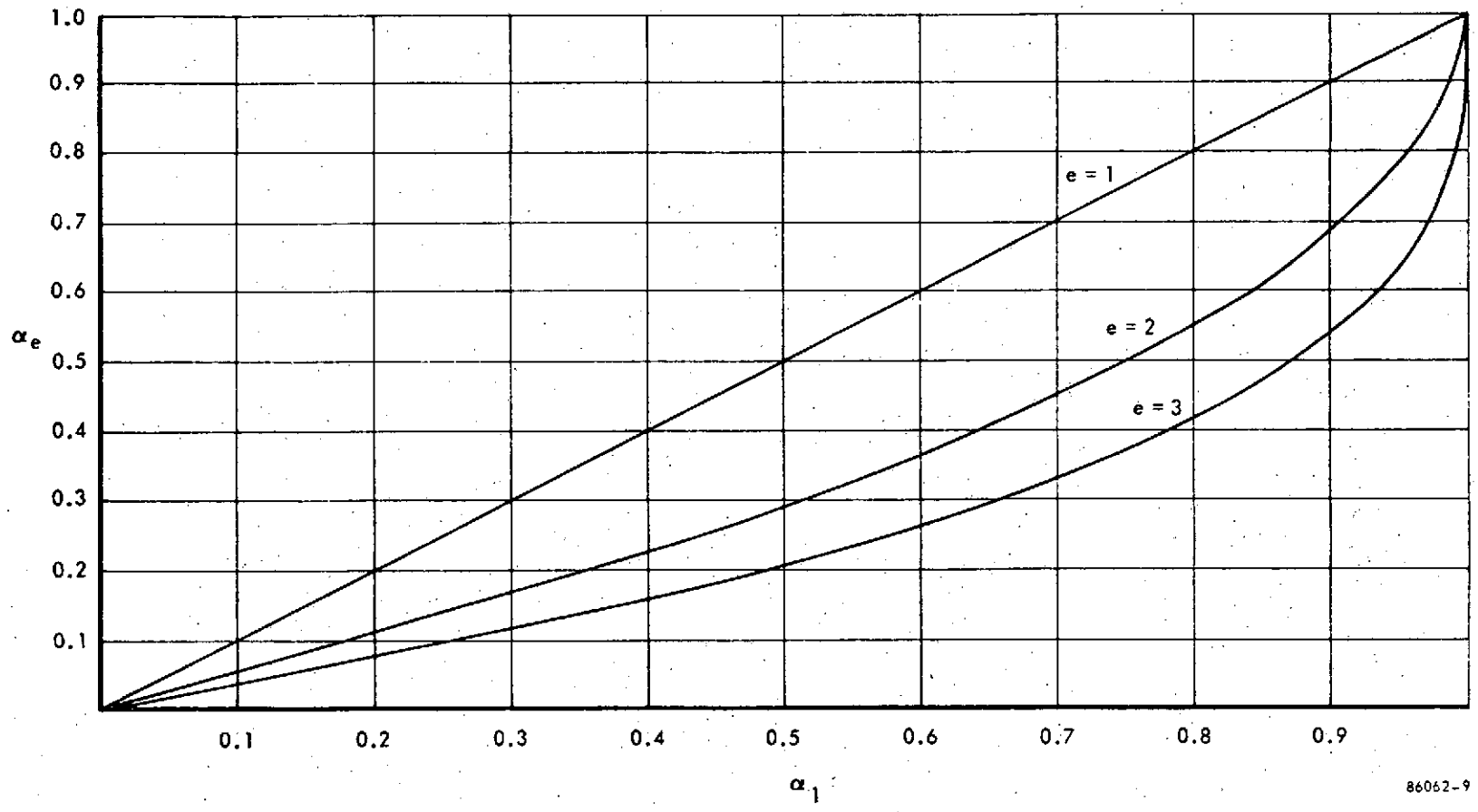



Figure 8.1.2. Equivalency of Smoothing Constants for Three Orders of Recursive Smoothing

serves the same function as a classical filter, that is, it reduces the potential data rate to remove the effects of rapidly fluctuating interference. Also, by combining N decisions before an alarm is generated, the alarm is delayed by at least N sample periods. This is a significant contrast to the predecision filters where the step response represented the worst case response. Here, step response represents the best possible response under any conditions.

Except for the contrast in step response, post-decision filtering strives to accomplish the same things as predecision filtering. It offers potential advantages in simplicity of implementation, particularly where several variables are employed in making the decision. The following paragraphs will discuss and compare the performance capabilities of several implementations of post-decision filters. So that the various implementations may be compared, Section 8.2.1 develops a performance criterion. Section 8.2.2 through 8.2.4 describe three implementations of post-decision filters.

### 8.2.1 Post-decision Performance Criterion

There are basically two areas for performance comparison. The first has to do with implementation of the filter. Since the filtering is a software operation this reduces to determination of CPU computing power and storage requirements for the various filtering techniques.

The second area of comparison is functional. How much improvement is given by the filter? There is a wide range of performance comparison possibilities and unless a common measure can be achieved for the general case, it will be quite difficult to trade filter implementations. For this reason a single parameter comparison that allows a quantitative discussion of performance has been selected. This criterion is the number of samples from an initial state until an alarm has occurred with 50% probability. The criterion assumes identical and independent statistics on each decision. It is a function only of the input probability of an error indication and the filter implementation.

If the filter acts on discrete batches of samples and makes an alarm decision, the probability of at least one alarm after I batches is one minus the probability that no alarms have occurred.

$$P_{aI} = 1-(1-P_a)^I$$

where $P_a$ is the probability of an alarm on any batch. For

$P_{aI}$ of 0.5, the number of batches for 0.5 probability of detection is

$$I = \frac{\ln 0.5}{\ln(1-P_a)}$$

We define $P_e$ as the probability that a single decision sample (input to the filter) is an error or out-limit indication. Note that for the case of N equal unity, i.e., no filter, $P_a$ is identically $P_e$.

The performance of each implementation discussed in the following sections will be assessed based on $P_e$ and the average number of input samples (average length of a run) until a 0.5 probability of alarm is reached. We shall call this number of samples $N_s$.

8.2.2 Up/Down Counter

This filter is a simple up/down counter which increases on an error decision and decreases on no-error decision. If it reaches a level N, an alarm is produced. In general, a no-error decision will decrease the count by K and an error decision will increase the count by J. The count is not allowed to go negative.

The software requirement for each decision is 2 comparisons and an increment or an initialization.

Performance can be determined by assigning each count of the counter a state and looking at state transition probabilities from sample to sample. The probability of being at count i on decision n is

$$P_{o,n} = (1-P_e) \sum_{l=0}^{K} P_{l,n-1}$$

$$P_{i,n} = P_{i+K,n-1} (1-P_e), \quad 1 \leq i < J$$

$$P_{i,n} = P_{i+K,n-1} (1-P_e) + P_{i-J,n-1} P_e, \quad J \leq i \leq N-K-1$$

$$P_{i,n} = P_{i-J,n-1} P_e, \quad N-K \leq i < N$$

$$P_{N,n} = P_{N-J,n-1} P_e + P_{N,n-1}$$

The performance of this filter is shown in Figure 8.2.2 for J=K=1 and N=1,4,8. The curve for N=1 represents

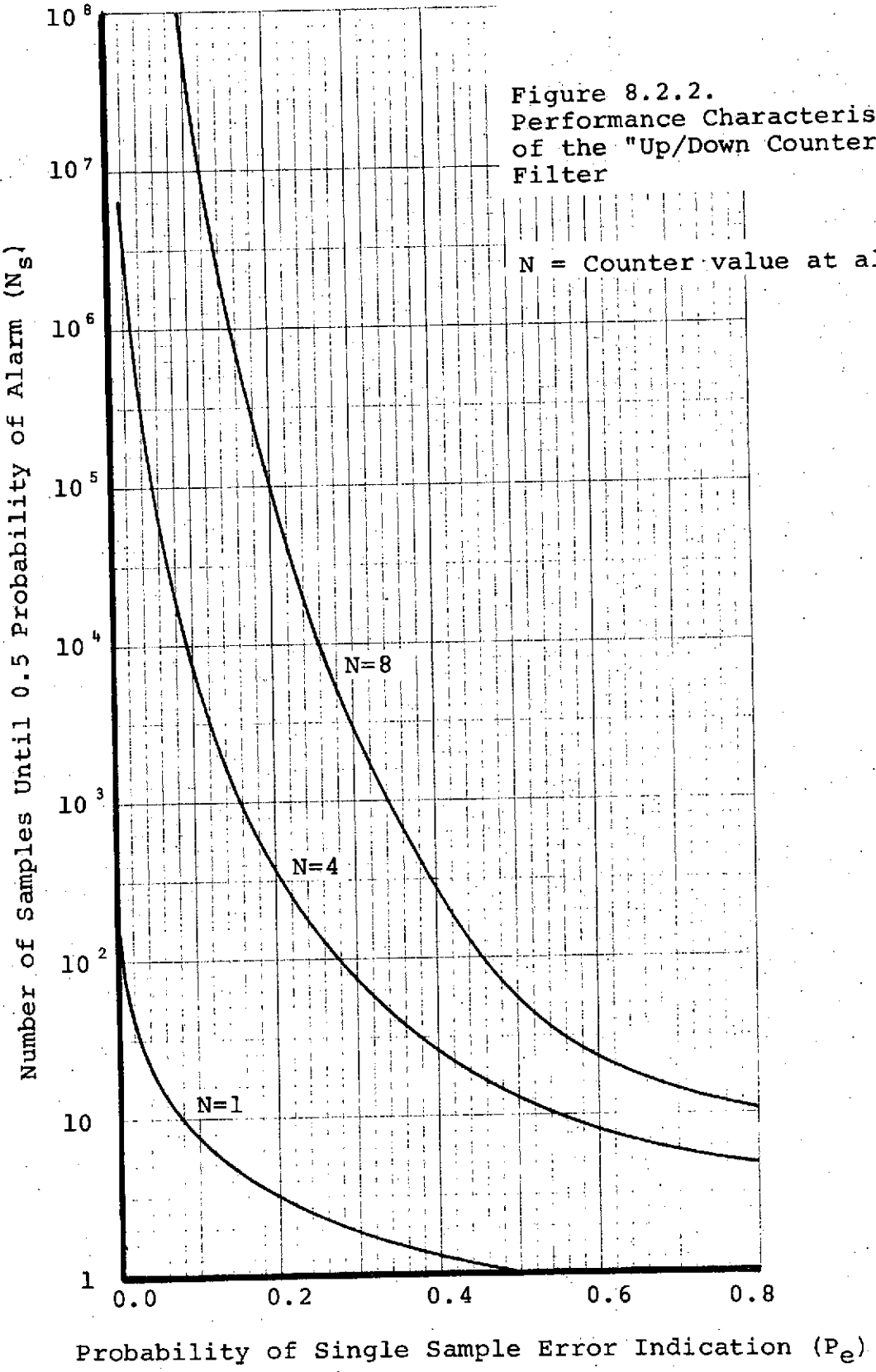



Figure 8.2.2.
Performance Characteristics of the "Up/Down Counter" Filter

N = Counter value at alarm

the alarm probability with no filter. Note that with no filter and $P_e = 0.1$, the chances are 50-50 of getting an alarm in just nine samples.

The minimum response time for each filter is the value of N.

8.2.3 M Out of N Counter

A flow diagram for this filter is shown in Figure 8.2.3-1. N decisions are counted. If there are M or more error decisions, the error alarm is generated. The process requires 1 increment and 2 comparisons for a no error decision and an additional increment and comparison if an error decision is made. Since this worst case should be included in a budget, implementation requires 2 increments and 3 comparisons.

Note that the technique described here takes a block of N samples and makes a decision. A technique requiring M out of the last N decisions could also be implemented but requires considerable more processing and memory of the last N decisions.

The probability of their being M or more error decisions in N samples is

$$P_a = \sum_{i=M}^{N} \frac{N!}{i!(N-i)!} P_e^i (1-P_e)^{N-i}$$

The number of samples is NI plus, on the average, N/2 at initialization from a previous sequence.

The performance of this filter is plotted in Figure 8.2.3-2 for four combinations of N and M. Since this filter is operating on batches, its step response is not necessarily N. It is reasonable to assume that a step input will occur (on the average) in the middle of a batch. The step response is then,

$$\begin{aligned} &M, && 1 \leq M \leq N/2 \\ &M+N/2, && N/2 < M \leq N \end{aligned}$$

Comparing these results to those of the up/down counter, it is seen that the up/down counter has a smaller step response for any given value of the performance criterion, $N_s$ and N.

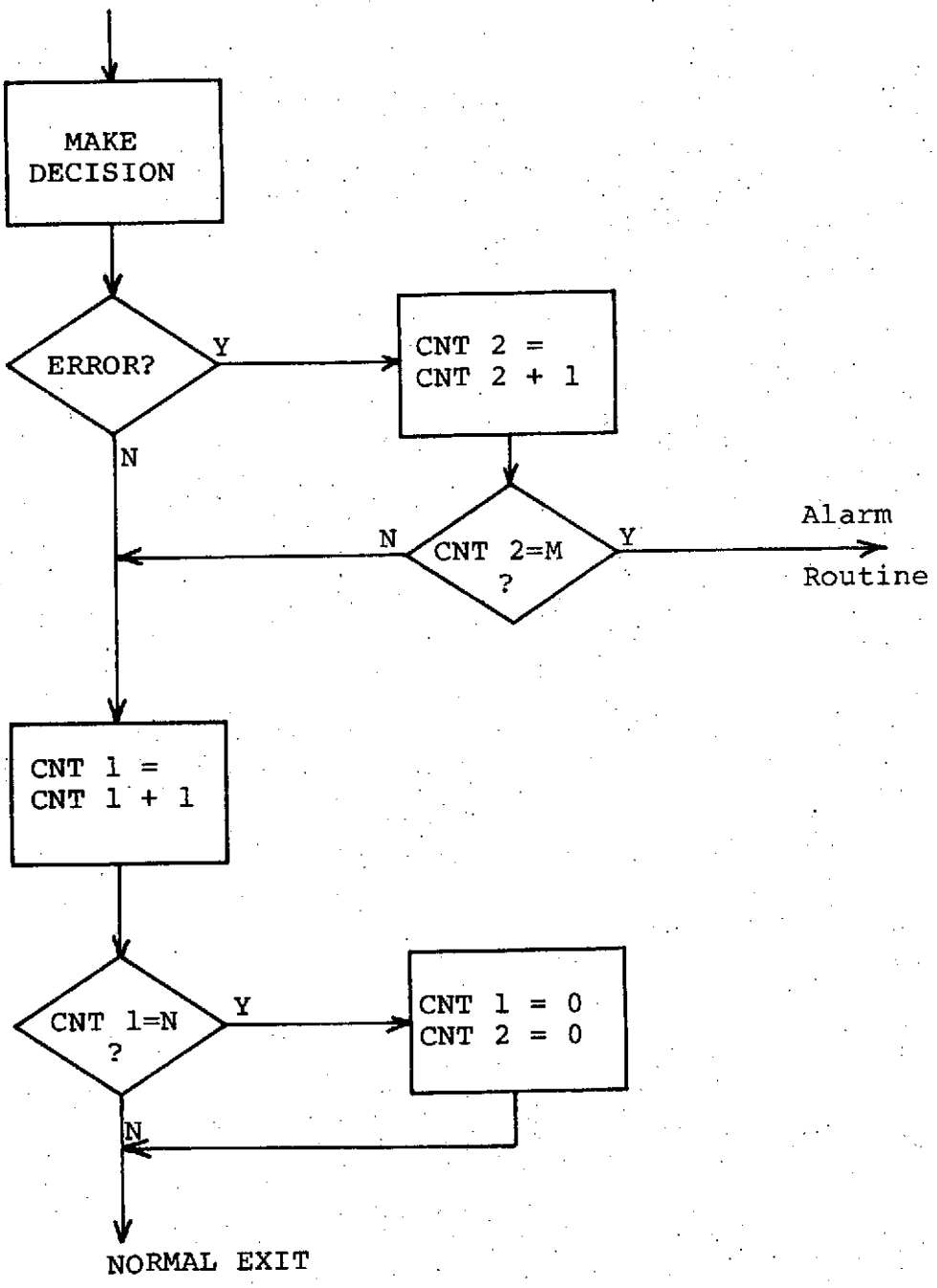



Figure 8.2.3-1. M Out of N Flow Diagram

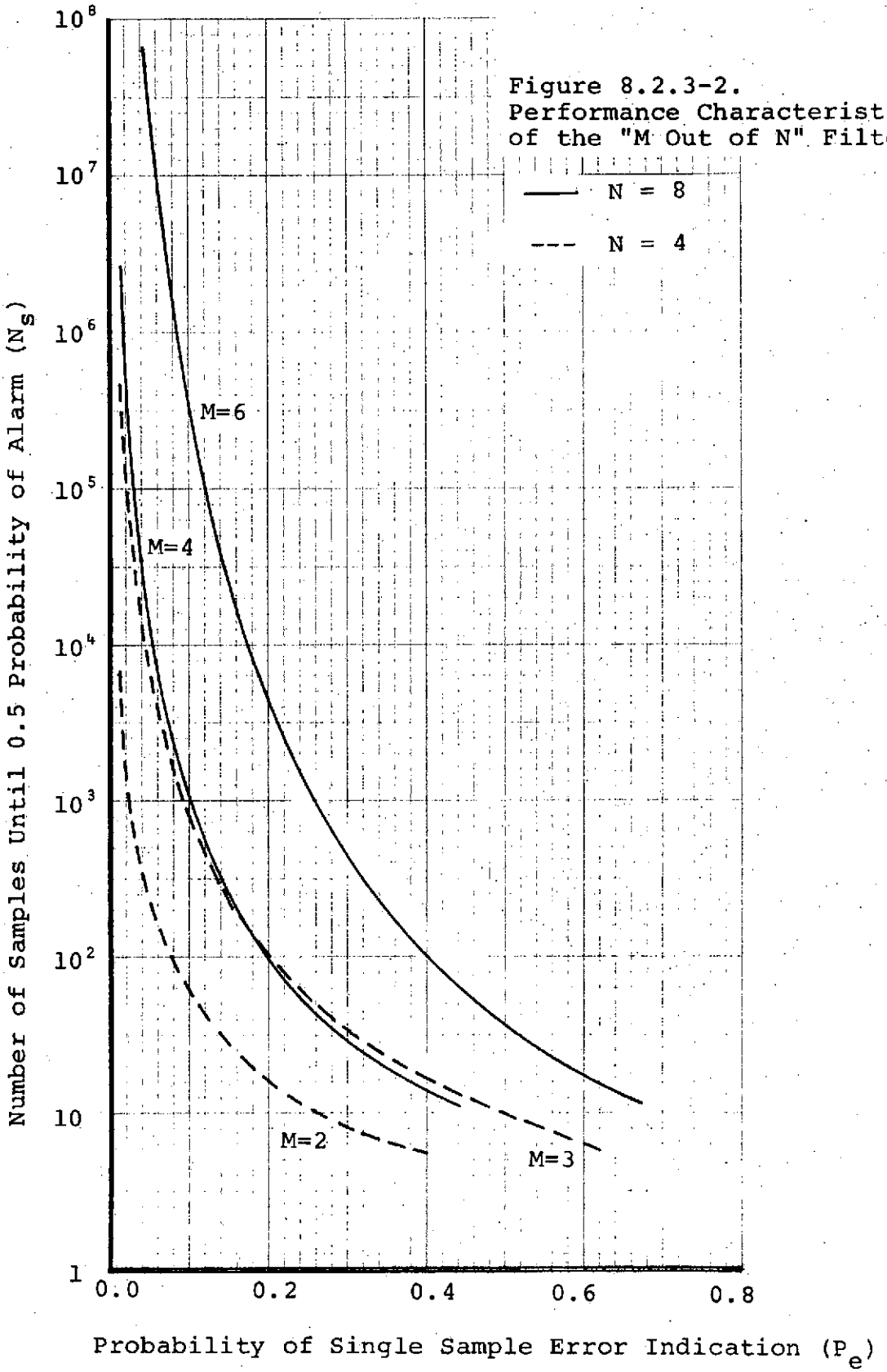



Figure 8.2.3-2.
Performance Characteristics
of the "M Out of N" Filter

### 8.2.4 Run of N Filter

In the limit of the M out of N decisions there is the N out of N case. This is given special treatment here because its implementation can be easily carried out on the last N instead of a block of N giving a faster response time. The case is also a special case of the up/down counter where J is 1 and K is equal to N.

The filter operates by counting consecutive error decisions. An alarm is sounded when N consuective error decisions are counted. Any single non-error decision which occurs before N is reached will cause the count to be reset to zero. The filter requires 1 comparison and an initialization if no error is indicated and 2 comparisons and an increment if an error is indicated so that implementation for worst case computations is identical to the up/down counter.

The probability of the first run of N errors occurring on sample n is the probability that no run has occurred by sample n-N-1 followed by a non-error decision, followed by N consecutive error indications. This can be formulated as

$$P_n = P_e^N(1-P_e)^{n-N-1}\left(1-\sum_{i=0}^{n-N-1} P_i\right)$$

where the last two factors account for the occurrence of no run. We seek that value of n for which $P_n = 0.5$.

The performance of this filter is plotted in Figure 8.2.4 for N=4,8. It can be seen that the false alarm immunity for this filter is greater than either of the others considered. Note also, that for very small values of $P_e$ there is little difference between the performance of this filter and the up/down counter.

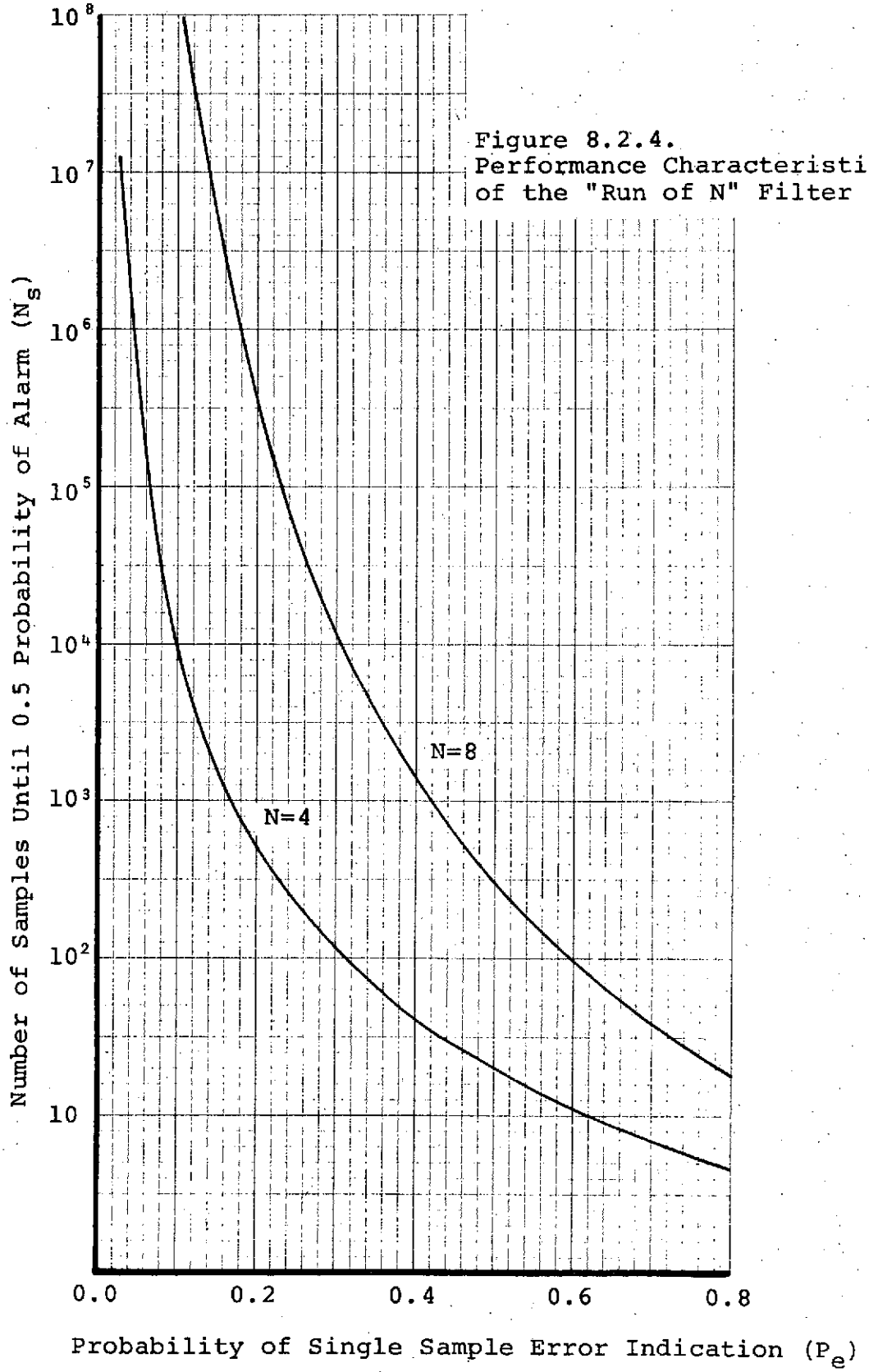



Figure 8.2.4.
Performance Characteristics
of the "Run of N" Filter

# 9.0 SUMMARY AND CONCLUSIONS

This volume has developed a methodology for designing automated performance verification and provided techniques and concepts in support of that methodology. A design procedure has been advanced as well as a formal structure for the verification process. Emphasis has been placed on verifying functions while the function is performing its operational role.

A procedure for selecting items to be verified as well as a set of criteria for determining functions for which verification is most advantageous have been presented. A set of techniques for verifying performance and establishing measures of performance are described along with their implementation considerations. Finally, a survey of smoothing or false alarm avoidance techniques is presented. The techniques have been supported by design aids and performance characteristics.

The methodology has been applied to an example using the Orbiter Hydraulics System.

It was concluded that operations and redundancy were the two most significant factors influencing the selection of items for which status must be determined. Whether an item should be verified in real- or deferred-time is primarily determined by whether:

- it is on-line or off-line
- it is very reliable
- real-time verification is warranted from the standpoint of cost, confidence in the results, ability of the crew to manually verify operation

It was pointed out that attempts to reduce false alarms always increase the delay to fault notification. Use of a predecision smoother guarantees a maximum value to this delay, whereas post-decision cannot guarantee such a maximum.

In the discussion of verification techniques it was noted that achieving performance verification on a real-time basis presents a greater challenge than designing a functional lab or bench test. The development and extraction of meaningful measures of performance is the key to successful automated performance verification. The designer of the item being verified must play a significant role in establishing these measures. It is quite likely that remote performance measure processing, to include analog implementations, will be required for digital computer versions of automated performance verification.

# APPENDIX A

# SYSTEMS MANAGEMENT DESIGN EXAMPLE FOR SHUTTLE HYDRAULIC SYSTEM

## A1.0 INTRODUCTION

The purpose of this appendix is to illustrate the application of the developed performance verification concepts to a typical subsystem of the Orbiter. The Hydraulic System was chosen based on its ease of definition which, on the other hand, demonstrates some of the complex concepts involved in the implementation of system performance verification. This appendix includes a brief description of the Orbiter Hydraulic System including its modes of operation over the mission phases. This discussion is followed by a detailed discussion of a performance verification design for the Hydraulic System in Section A3.0. Relating these results to the Orbiter System Management Function under development at this writing, the design described herein reflects an implementation of Orbiter functional path fault detection.

## A2.0 ORBITER HYDRAULIC SYSTEM DESCRIPTION

The purpose of this section is to describe the Hydraulic System as applied to the performance verification process. The Orbiter Hydraulic System is used to provide hydraulic power during the boost, re-entry, and landing phases of flight. Redundancy is provided in the Hydraulic System by use of three (3) independent hydraulic subsystems. Hydraulic power is provided to the appropriate orbiter functions through automatic switching valves that switch to alternate hydraulic systems in the event of failure of the primary Hydraulic System. (For the purposes of this analysis, the three hydraulic systems which comprise the hydraulic power source for Orbiter will be referred to as hydraulic systems and not as hydraulic subsystems. This should not be confused with the total hydraulic system which is made up of the the three hydraulic systems.)

Table A2-1 presents the shuttle system functions involving the hydraulic systems. Hydraulic power is required during boost and from re-entry through landing at which time full hydraulic pressure is required. The remaining portion of the mission (except during re-entry checkout) involves essentially housekeeping functions for the Hydraulic System during which time the hydraulic fluid is circulated periodically through the system to maintain the fluid to within its operating temperature range.

The detailed description of the Hydraulic System performance and components are found in Reference 1 and 2.

The Hydraulic Power System incorporates functional redundancy provisions to ensure that operation of hydraulic-driven components can continue after specified failure. With the exception of nosewheel steering and strut actuators, redundancy is obtained by switching valves located at the actuators which provide the capability to automatically select the hydraulic system supplying hydraulic power to that actuator in the event of a pressure drop in the currently used hydraulic system.* This pressure drop is sensed at the switch valve. Each switch valve is initialized to supply hydraulic pressure by one hydraulic system which is designated as the active system. The other systems are connected to the actuators through the switching valve and are designated Standby 1 (S1) or Standby 2 (S2) systems. In the event of a failure of the Active (A) System, the switching valve automatically switches over to the S1 system. In the event that system (S1) fails, changeover to the S2 system is automatic.

---

*The hydraulic components in the subsystems accommodating nosewheel steering and strut actuators cannot switch to alternate hydraulic power sources. They remain fixed to one hydraulic system.

Table A2-1. Orbiter Hydraulic System Mission Phase Modes

| Mission Phase | Function | Hydraulic System Function |
|---|---|---|
| Boost | TVC controls for main engine<br>Fuel control for main engine | Hydraulic Power |
| On-Orbit | Maintenance of fluid temperature<br><br>Re-entry checkout | Circulate fluid and temperature control<br><br>Hydraulic power |
| De-Orbit/Re-Entry | Flight Control<br><br>Elevons<br>Body Flap<br>Speed Brake/Rudder | Hydraulic power |
| Landing/Deceleration | Landing gear/brakes<br><br>Strut Actuators<br>Uplocks<br>Brakes<br>Nosewheel Steering<br><br>Fluid cooling<br>System checkout | Hydraulic power<br><br><br><br><br><br>Cool fluid<br>Hydraulic power |

Table A2-2 presents the functions and the corresponding hydraulic systems with designations as Active (A), Standby 1 (S1) or Standby 2 (S2). It is observed that two functions, the body flap and speed brake/rudder employ all 3 hydraulic systems as active hydraulic power sources. This is accomplished by each system driving a motor which is mechanically summed through a differential to drive the aerodynamic surfaces. The system is designed such that 50% power is utilized for each system to achieve the maximum actuator rate. Therefore, loss of one system can be totally compensated for by the other two. Loss of two systems would require application of the priority control system in which the aero surfaces are rate-limited after 2 hydraulic system failures. Similarly, after touchdown, two active hydraulic systems are active for braking.

During the on-orbit phase of flight, the main function of the hydraulic system is to remain operable. Electrically driven circulation pumps move hydraulic fluid through the system and (as temperature demands) a fluid heater to maintain the fluid temperature to within its operating ranges. Each pump is operated for 20 minutes on the hour and, in this manner, all 3 pumps may be run 1/3 of the time each hour, cycling one to another. This operation reduces electrical power drain.

Table A2-2. Assignment of Hydraulic Systems To Orbiter Functions

| Function | | System 1 | System 2 | System 3 |
|---|---|---|---|---|
| TVC, Main Engines | | | | |
| Control Actuators | | | | |
| Engine 1 Yaw | | A | S2 | S1 |
| Engine 1 Pitch | | A | S1 | S2 |
| Engine 2 Yaw | | S2 | A | S1 |
| Engine 2 Pitch | | S1 | A | S2 |
| Engine 3 Yaw | | S2 | S1 | A |
| Engine 3 Pitch | | S1 | S2 | A |
| Main Engine Control | | | | |
| #1 Controls | | S1 | A | - |
| #2 Controls | | - | S1 | A |
| #3 Controls | | S1 | - | A |
| Flight Controls | | | | |
| Elevons | Rt Outbd | A | S1 | S2 |
| | Rt Inbd | S1 | S2 | A |
| | Lt Outbd | S2 | S1 | A |
| | Lt Inbd | S1 | A | S2 |
| Body Flap | Motor Pwr | A | A | A |
| Speed Brake/ Rudder | Motor Pwr | A | A | A |
| | Logic | S2 | S1 | A |
| Deceleration & Landing Strut Actuators | | A | - | - |
| Uplocks | | S2 | A | S1 |
| Brakes | Rt Outbd | A | A | S1 |
| | Rt Inbd | A | A | S1 |
| | Lt Outbd | A | A | S1 |
| | Lt Inbd | A | A | S1 |
| Nosewheel Steering | | A | - | - |

During on-orbit & landing — circ. pump for heating or cooling fluid.

## A3.0 PERFORMANCE VERIFICATION DESIGN DEVELOPMENT

System performance verification requires an organized and systematic approach to the design of an efficient performance verification system. Because of the complexity of a Shuttle system, the objective is to minimize the amount of data reported to the crew by the performance monitor and yet to still maintain a high degree of capability in verifying performance of the subsystem. Stated in terms of the System Management problem, reliable performance data should be reported at the level for which the crew is capable of taking action. This, then, equates to reporting Orbiter faults to the functional path level. As pointed out in the body of this volume, by definition of items to be verified (IBV's) and by choice of appropriate measures of performance (MOP's) of each, the performance verification system is established. As the analysis progresses, it should become obvious that an IBV and a functional path (in Orbiter design terminology) are one and the same.

One of the key elements in the performance verification function is the definition of the system bounds to ensure that the items being verified (IBV's) and measures of performance (MOP's) are truly a measure of performance (and verification) of the system and not of another system located functionally upstream (input) or downstream (after output) of the system in question. As a result, great care was taken to define the bounds of the hydraulic system.

This section includes a discussion of the IBV selection for this problem, their bounds and limits; a description of the performance verification system requirements, the definition of measures of performance and a discussion of the performance verification system decision rules.

### A3.1 Identification of IBV's

As discussed earlier, one of the most important parts of the performance verification process is the proper definition of the items to be verified. Care must be taken to ensure that the IBV presents a level consistent with providing a good indication of level of performance without providing redundant information.

Three IBV's are defined in the hydraulic system. They are the three independent hydraulic systems used to furnish hydraulic power to the entire Orbiter vehicle. These are chosen because they represent the smallest subdivision of the hydraulic power system that can fail and directly relate to failure of a system which limits hydraulic power capability (or at least redundancy). Failure of any component of the hydraulic system would result in an effective failure of the hydraulic system, and therefore, would provide useless

information since no means of repair have been provided. For example, failure of the APU or main pump results in loss of hydraulic pressure and, therefore, loss of the hydraulic system,

Figure A3.1 shows the Hydraulic System Functional Flow Diagram (3 included in the Hydraulic Power System) for the Orbiter System. The bounds of the defined IBV are indicated. As illustrated, included in the IBV are all the elements for power generation (APU, Automatic Pressure Control, main pump with electric depressurization valve for start), fluid distribution (lines and part of the switching valves) and fluid property maintenance (filter, heater, cooler, circulation pump). The bounds of the IBV cross the input on-off signals for the pumps and APU and the lines going to the actuators and motors. The switching valves (to switch to alternate subsystems in the event of loss of power on one subsystem) are partially included within the bounds of the IBV since they are used in the power distribution function but their automatic selection capability is not part of the hydraulic system function. The bounds cut the lines going to the actuator and motors because a failure in that line would disable the actuator but not the Hydraulic System itself. The one exception to this is the development of a leak in the actuator line. If a leak develops in the control valve, actuator, or in the lines to the actuator, it may lead to failure in the system due to loss of hydraulic fluid. This type of failure could be catastrophic since, unless isolated, that failure would first cause the primary Hydraulic System to fail; the switch valve would then switch to the first standby and proceed to fail it until all three hydraulic systems were drained of hydraulic fluid.

It should be noted that the pump pressure regulator package has been included as part of the IBV. This is necessary since its performance is essential to IBV operation. This is a good example of the differences that can occur between performance verification partitions and subsystem design partitions.

The choice of the three IBV's as the three independent hydraulic systems making up the total hydraulic power system on the Orbiter is consistent with the requirements that the IBV be as large as the significance or consequence of failure. IBV's at a level lower than that would have the same impact as this and would require more IBV reporting. Furthermore, once a failure is indicated on this IBV, the System Management (SM) system can follow with a review of component status to isolate a failure and/or to assess the degree of failure.

Since the distribution of power in the hydraulic system is performed automatically from one system to another

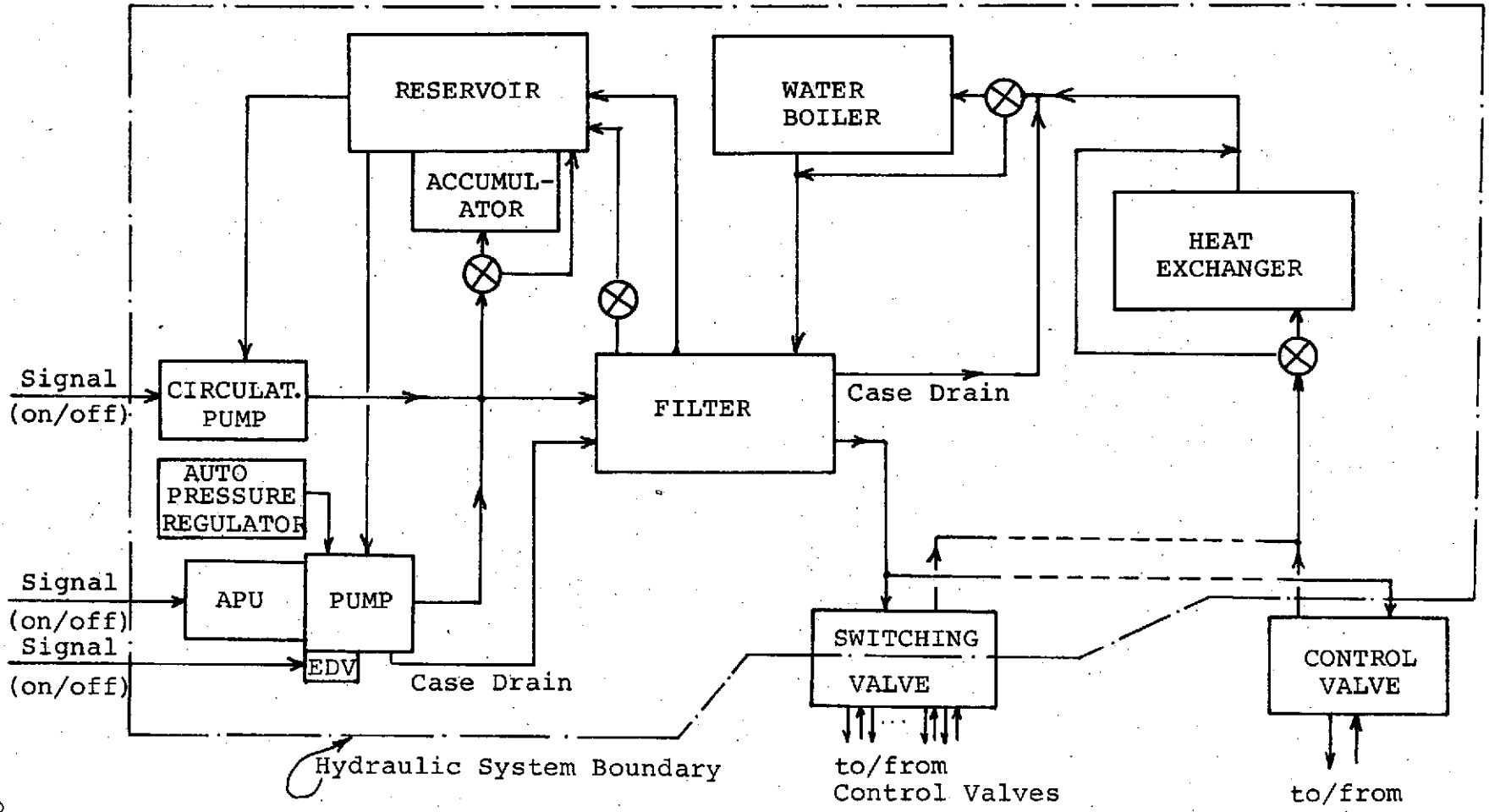



Figure A3.1. Hydraulic System Functional Flow Diagram

when a switch valve senses loss of pressure, the Shuttle crew does not need to take any direct action in the event of a single IBV failure (except possibly to turn off that hydraulic system). However, the crew must be aware of loss of the system since it limits the total hydraulic system capability from a redundancy standpoint as well as, in the case of two hydraulic system failures, providing less hydraulic power for the body flap and speed brake/rudder during aerodynamic control. In short, Oribter dynamics change with the number of failed hydraulic systems and the crew will likely alter decisions based on hydraulic system status information.

A3.2 Identification of the Performance Verification Requirements

Once the IBV's have been established, performance requirements on the performance verification system must be defined. Typically, such requirements include the following:

- Response time requirements for the Performance System — timeliness requirements for failure data on the IBV.
- Accuracy and confidence requirements — i.e., what is the tolerance for false alarm or misses?
- Definition of different modes of operation of the IBV and how the Performance Verification System responds to those differing modes.
- Frequency for verification of the IBV function or subsystem.
- Performance criteria on which to base measurement of operational integrity — the relationship of the IBV output to the required performance indications or the relation of the rationale of the IBV performance with respect to means of identifying its performance.

In short, specifications for the design of the performance verification system are being defined and related to system performance. The important point here is that the IBV performance requirements and use dictates the requirements for the performance verification system design.

The performance verification system requirements are heavily related to the significance of the IBV and the accuracy and timeliness requirement for reporting the status of that IBV. It must be adaptive to accommodate the requirements during the different mission phases.

There are two distinct modes of hydraulic system operation during a Shuttle mission. One is when it is operating to provide full hydraulic power during the boost, re-entry checkout (on-orbit), and during re-entry through landing phase of flight. The second mode is the on-orbit mode where the hydraulics are not being used and the only function is the maintenance of hydraulic fluid temperature. During the first mode of operation, the performance verification system needs to report continuously, the status of the hydraulic system. However, during the second mode, the system is not in use and therefore cannot be truly tested for failure. Therefore, during the second mode of operation (on-orbit) no performance verification data is required. However, the measurements associated with the circulation system and fluid temperature maintenance should be available to the Caution and Warning system so that if the temperature gets too cold and/or too hot, and/or the fluid level in the reservoir gets too low, a warning would be signaled. (This is not a function of fault detection/performance verification).

The hydraulic system, with its redundancy and associated GN&C automatic switching capability, is somewhat forgiving in the area of response time and accuracy of performance verification reporting. Therefore, a response time (although not known at this time) should be on the order of a few seconds to inform the crew of failure of the particular hydraulic systems. Similarly, with only one failure, the accuracy of the data can be suspect, however, when another hydraulic system (IBV) fails, the burden may be placed on the crew to verify the accuracy of the performance verification system, at which time alternate sensors and subsystem status can be monitored.

How the status of redundant hydraulic systems is reported could be a simple tally of the number of operative systems (IBV's), e.g., 2 of 3 are operative. Since all are identical, this approach seems attractive due to a large amount of information contained in the single statement. Unfortunately, the effects of IBV failures are not identical. As indicated in Section A2.0, the loss of Hydraulic System 1 results in a degraded performance not realized by loss of System 2 or 3 (i.e., loss of N/W steering and gear actuators). Thus, it will be necessary to display the status of each IBV.

The performance criterion for the hydraulic system is essentially to provide hydraulic power (pressure) during its operating mode (and on-orbit checkout). During the on-orbit mode, there is no real performance output, therefore, no performance criteria is indicated except those inputs to the Caution and Warning system whose function is to annunciate potential problems.

## A3.3 Identifying Measures of Performance

The primary function of the hydraulic system is to provide hydraulic pressure for use by numerous actuators on the Orbiter System. Therefore, the hydraulic pressure is the most important measure of performance. (This is particularly true since the hydraulic pump is pressure regulated.) Two other measures of performance are the level of the fluid in the reservoir and temperature of the hydraulic fluid. These two measures provide indication of the performance status of the system. The reservoir fluid level is an indication of system leaks and of impending loss of pressure resulting from loss of fluid whereas the fluid temperature is important to ensure proper operation of the hydraulic system.

All three measures of performance:

Pressure
Fluid Level in the Reservoir
Fluid Temperature

satisfy the guidelines and are characterized as follows:

Quantifiable
Identifiable Range of Limit
Representative of the Function Being Performed
Responsive and Not Volatile

The only measure of performance which is volatile is the pressure MOP. Although the system is designed to respond very quickly to provide a constant pressure during actuator dynamic operation, the pressure transient could report a failure for a moment or two during that transient period. This will be discussed in more detail in the next section.

## A3.4 Identifying Decision Rules

At this point in the development of the performance verification, the ground work has been established by the identification of system performance requirements, identification of IBV's which are indicative of system performance, definition of performance verification requirements for the system, and definition of the measures of performance (MOP's) for each IBV (in this case, each of the IBV's are identical). The purpose of this section is to develop the rules for defining system status for the IBV's utilizing the MOP's identified above.

Hydraulic pressure is the first MOP defined for the hydraulic system. During use of the hydraulic system, i.e., dynamic flight, the pressure of approximately 3000 psi must be maintained. If the pressure departs appreciably from this valve, then a fault is reported. The high and low pressure

bounds for the ALT mission have been described to be 2850 psia to 3050 psia (Ref. 1). Consideration must be given to two things here:

1) What is the variation in hydraulic pressure when all the actuators and motors are being used?

2) What is the range of pressure required for nominal system performance?

It is understood that the hydraulic pump is controlled to provide constant pressure throughout the mission and that under minimal hydraulic power usage, these limits will not be exceeded.

However, if a large variation in pressure were experienced during maximum power usage and if the power requirement variation is smaller than that experienced during the transient, a provision to account for the transient in the performance verification system must be provided. This is exemplified in Figure A3.4. One observes that during the transient condition, the pressure drops below the pressure envelope, however, for only a short period of time. This would report a fault condition which, for an instant may be true due to pressure being below the required threshold; however, as far as the performance of the Hydraulic system is concerned, everything would be operating according to specification. In this area, the MOP should be defined to accommodate the transient low pressure condition.* A possibility is to restate the MOP in terms of RMS pressure or pressure averaged of a running, say, 100 msec window. This quantity would then be tested against the stated bounds. This concept can be expanded to provide a means of reducing the pressure fault limits for the hydraulic system if the transient conditions are accounted for.

The second measure of performance is the level of fluid in the reservoir as an indication of nominal performance of the system. There is some question as to the role reservoir fluid level plays in the Performance Verification system. It has a tendency to be related to caution and warning in reporting a low fluid level. The measure may not be required. It has been identified to compensate for rapid pressure fluctuations resulting from pump cavitation when fluid is low. The average pressure MOP could take care of this detection also.

*This MOP restatement should not be confused with false alarm avoidance or smoothing. A different MOP is being defined and calculated based on IBV performance. False alarm avoidance is a separate problem.

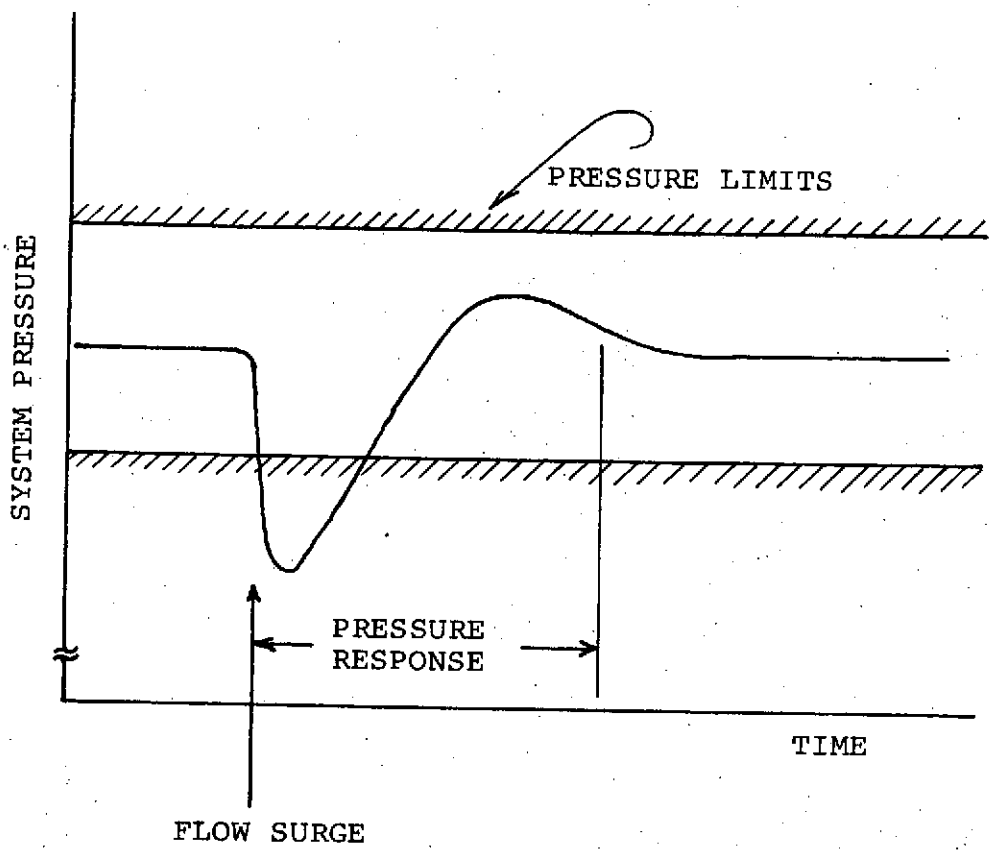



Figure A3.4. Hydraulic System Response to Step Load

The third measure of performance is fluid temperature in the system. The fault detection or performance verification for this MOP presents a failure if the temperature of the fluid falls out of the design temperature bounds which are (TBD) to 250°F. The MOP has been defined to accommodate hydraulic dynamic variations due to variations in viscosity. It too, may not be necessary if the pressure MOP is properly defined.

In summary, the criterion for reporting failure on the Hydraulic System during power operation include the three measures of performance:

| | |
|---|---|
| Pressure | - Within specific pressure limits with provisions in the performance verification to account for the transient conditions. |
| Reservoir Fluid Level | - Report failure when it drops below 5% — consideration should be given to replacing this MOP with a more conclusive pressure MOP. |
| Fluid Temperature | - Indicating a failure mode in the fluid if the temperature of the fluid exceeds the bounds of +250°F and -(TBD)°F. Consideration should be given to replacing this MOP with a more conclusive pressure MOP. |

No performance verification is employed in the on-orbit mission phase.

REFERENCES:


1. "Orbiter 101 Subsystem Simulation Requirements for Performance Monitor Functional Simulator, Hydraulics," L.E.C., No. LEC-4130, Aug. 1974.

2. "Orbiter 101 Subsystem Simulation Requirements for Performance Monitor Functional Simulator, Landing and Deceleration," L.E.C., No. LEC-4404, Sept. 1974.

# APPENDIX B

# MOP EXTRACTION TECHNIQUES

## B1.0 INTRODUCTION

The purpose of this appendix is to provide descriptions and some discussion of several typical MOP Extraction Techniques. These techniques, along with numerical and analog descriptions of their operation, are summarized in Table B1.0. As may be seen, they are partitioned into broad classes on the basis of number of IBV's involved and source of the information used in MOP calculations. These classes are defined in Figure B1.0. In all cases, the assumption is that the information used in the calculations is provided as a time series of numbers representing the IBV input or output signal, as the case may be.

Throughout this appendix {X} denotes the set of numbers derived directly from the IBV. {Y} denotes the set of determined MOP's for the IBV.

Altogether ten MOP techniques are covered. Of these, five are MOP Class 1, two are MOP Class 2, and three are MOP Class 3.

Each of the ten techniques is described and discussed below. Generally, the technique name denotes a whole collection of more specific techniques. For example, "spectral analysis" denotes many techniques which share the characteristic frequency domain measures obtained from a time series; but the number of practical implementations of the Discrete Fourier Transform is large, and the selection of an optimal technique depends strongly on both the specific time series data and other implementation constraints (e.g., computer run time, memory size, and word size restrictions).

TABLE B-1 SOME TYPICAL MOP EXTRACTION VALUES

| NUMBER | SUBCLASS NAME | MOP CLASS# | DESCRIPTION OF GENERAL CASE (DISCRETE VERSION) | TYPICAL EXAMPLE (CONTINUOUS TIME VERSION) |
|---|---|---|---|---|
| 1 | SEQUENTIAL VALUE CHECK | 1 | $\{Y_{ilm}\} = f\{X_{ijk}\}$, f any function | $Y(t) = \frac{d}{dt}(X(t)) - K$ <br> K = CONSTANT |
| 2 | NON-SEQUENTIAL VALUE CHECK | 1 | $\{Y_{ilm}\} = f\{X_{ijk}\}$, f any function for which the value doesn't depend on the order of the $\{X_{ijk}\}$ | $Y(t) = X(t) - K$ <br> K = CONSTANT |
| 3 | CODING | 1 | $X_{ijk} = d_1, d_2, \cdots d_n$ , $d_i = 1$ or 0 <br> $\{Y_{ilm}\} = h\{X_{ijk}\}$, h some Boolean function of digits of $\{X_{ijk}\}$ | $Y = (d_1 \oplus d_2 \oplus \cdots \oplus d_{n-1} \oplus d_n)$ <br> $d_n$ = parity bit |
| 4 | STATISTICAL ANALYSIS | 1 | $\{Y_{ilm}\} = f\{X_{ijk}\}$, f some statistical function | $Y(t) = \left(\frac{1}{t}\int_0^t X(\eta)\,d\eta\right) - K$ <br> K = CONSTANT |
| 5 | SPECTRAL ANALYSIS | 1 | $\{Y_{ilm}\} = f\{X_{ijk}\}$, f some version of the discrete Fourier transform | $Y(f) = \left\lvert \int_{-\infty}^{\infty} X(t) e^{-j2\pi ft}\,dt \right\rvert$ <br> $Y(f_2) - K_1$ $\quad K_1$ = CONSTANT |
| 6 | INVERSE TRANSFORM | 2 | $\{Y_{ilm}\} = f(\{X_{ijk}\}, \{W_{ijk}\})$, f a discrete version of some inverse transform <br> X = IBV outputs, W = IBV inputs | $Y(t) = g^{-1}(X(t)) - W(t)$ <br> W(t) = input signal <br> $g^{-1}$ = inverse transform function |
| 7 | CORRELATION | 2 | $\{Y_{ilm}\} = f(\{X_{ijk}\}, \{W_{ijk}\})$, f a discrete version of some correlation function <br> X = IBV outputs, W = IBV inputs | $Y(t) = h_T(t) - h_R(t)$ $\quad h_T(t)$ = response function <br> $h_T(t) = \int_0^T W_T(\tau)\, x(t-\tau)\,d\tau$ $\quad W_T(t)$ = unit impulse function (pseudo-random noise) |
| 8 | COMPARE TWO | 3 | $\{Y_{1lm}\} = f(\{X_{1ijk}\})$, $\{Y_{2lm}\} = f(\{X_{2ijk}\})$, f = any function <br> $\{Z_{lm}\} = g(\{Y_{1,l,m}\}, \{Y_{2lm}\})$, g = some difference function | $Z(t) = Y_1(t) - Y_2(t)$ |
| 9 | CROSSPOWER SPECTRAL ANALYSIS | 3 | $Z = f(\{X_{1ijk}\}, \{X_{2ijk}\})$, f a discrete version of the coherence function | $Z = \overline{\alpha}^2$ = coherence function <br> $\overline{\alpha}^2 = \frac{\lvert G_{12}\rvert^2}{G_{11} G_{22}}$ <br> $G_{12}(f)$ = cross spectral density $= \int_{-\infty}^{\infty} R_{12}(\tau) e^{-j\omega\tau}\,d\tau$ <br> $R_{22}$ similar to $R_{11}$ <br> $G_{22}$ similar to $G_{11}$ <br> $R_{12}(t)$ = time cross correlation function $= \lim_{T\to\infty} \int_{-T}^{T} X_1(t+\tau) X_2(t)\,dt$ <br> $G_{11}(f)$ = power spectral density $= \int_{-\infty}^{\infty} R_{11}(\tau) e^{-j\omega\tau}\,d\tau$ <br> $R_{11}(t)$ = time auto correlation function $= \lim_{T\to\infty} \int_{-T}^{T} X_1(t-\tau) X_1(t)\,dt$ |
| 10 | VOTING | 3 | $\{Y_{1lm}\} = f(\{X_{1ijk}\})$, $\{Y_{2lm}\} = f(\{X_{2ijk}\})$, $\{Y_{3lm}\} = f\{X_{3jk}\}$ <br> $\{b_{1lm}\} = \{Y_{1lm} - K_{1lm}\}$, $\{b_{2lm}\} = \{Y_{2lm} - K_{2lm}\}$, $\{b_{3lm}\} = \{Y_{3lm} - K_{3lm}\}$ $\quad \{K_{ilm}\}$ = CONST. <br> $\{Z_{lm}\} = h(\{b_{1lm}\}, \{b_{2lm}\}, \{b_{3lm}\})$ | $Z = (b_1 b_2 \overline{b}_3 + b_1 \overline{b}_2 b_3 + \overline{b}_1 b_2 b_3)$ |

FOLDOUT FRAME 1



FOLDOUT FRAME 2

| CLASS # | DEFINITION |
|---|---|
| 1 | MOP calculations are based on information regarding the output(s) of a single IBV |
| 2 | MOP calculations are based on information regarding both the input(s) and the output(s) of a single IBV |
| 3 | MOP calculations are based on information regarding the outputs of multiple (two or more) similar IBV's, which share the same inputs |

Figure B1.0. Classes of MOP Extraction Techniques

## B2.0 CLASS 1 MOP'S

All Class 1 techniques have in common the fact that only extracted numbers from the output(s) of a single IBV are used for the MOP calculation. Thus only conditional status can be determined directly from these MOP's since nothing is known regarding IBV input status at decision time ("conditional" means "conditional on the state of the input").

### B2.1 Sequential Value Check Techniques

As shown in Table B1.0, the general form for this subclass of techniques is

$$\left|Y_{ilm}\right| = f\left(\left|X_{ijk}\right|\right)$$

where f describes some time-ordered function of the IBV data. If the time series of X is described by $X_i$ this function could be $Y = X_n - X_{n-1}$. Or, $Y = \max \{X_n, X_{n-1}\}$, $Y = X_n - X_{n-1} - K$. These techniques are quite similar to nonsequential value checks except that here, order of the samples is considered to be important. An example of a sequential value check technique is the case where the derivative of the IBV output signal is required as a measure of performance. Numerically, this amounts to determining the difference between two successive samples. This particular example is illustrated in Table B1.0.

### B2.2 Non-Sequential Value Check Techniques

These techniques are similar to those above, except that f is constrained to be a function for which the value is independent of the order of the $\left|X_{ijk}\right|$. In its simplest form, this technique may involve the calculations of a MOP based solely on the most recent value of the sequence $\left|X_{ijk}\right|$, and the MOP may be the difference between the value of $X_{ijk}$ and some fixed reference value. This simple MOP is quite appropriate for many verification purposes; for example, it is useful in verifying the correctness of the sampled output voltage of a power supply.

### B2.3 Coding Techniques

Coding techniques are generally applicable only when output signals from the IBV are in digital form. A typical example of the use of coding for performance verification is the use of a parity check in the output of a core memory in a computer system. In this case, the IBV is the core memory, and

incorrect parity is the $X_i$. The MOP (Y) is developed from the $X_i$ on the basis of number of parity errors committed either consecutively or over some period of time.* A different form of the coding technique is also commonly used to verify the correct operation of magnetic tape subsystems of a computer system, and the technique has been proposed for verification of the correct operation of the arithmetic unit in computer systems.

In a typical example of the use of a coding technique, the extracted numbers will be of the form

$$X_{ijk} = d_1, d_2, d_2, \ldots . d_n$$

where $d_i$ is a binary digit (0 or 1). Some of the digits ($d_i$) contain redundant information, which is used to allow detection of certain types of errors in the $X_{ijk}$. The detection system can be simple, requiring only the computation of the Boolean function

$$X_i = d_1 + d_2 + \ldots . + d_n$$

and $$Y = \Sigma(N \text{ consecutive } X_i)$$

as a MOP. In this case, if Y = N then the conditional status of the IBV is "bad"; otherwise it is "good."

In the magnetic tape application mentioned earlier, Boolean computations are commonly done on rows and columns of binary arrays formed from sequences of $|X_{ijk}|$, and the associated codes are often called block check codes.

In general there are many different types of codes, some capable of detecting and possibly correcting mulitple bit errors. They are all characterized by the fact that redundant bits are added to non-redundant bits in a controlled manner; thus certain types of subsequent distortions of the entire number (redundant plus nonredundant bits) due to IBV failures, communications noise, or any other cause, can be detected or possibly corrected. The detection and correction of errors is generally done independent of the value of the number before redundancy was added, and this is a unique and often desirable feature of the coding technique.

*Recall that MOP's are used as failure indicators, not error indicators. The fact that a parity error has occurred does not necessarily mean that a failure has occurred. Errors must be translated into failures.

## B2.4 Statistical Analysis Techniques

The statistical analysis MOP techniques can be described as

$$\left|Y_{ilm}\right| = f\left(\left|X_{ijk}\right|\right)$$

where f is some statistical function of the extracted numbers such as mean value, variance, RMS, etc. Computation of such functions is routine for a general purpose digital computer, and many algorithms are known and used for the more common functions. Thus there are no conceptual implementation difficulties associated with the technique; whether or not it is suitable depends mainly on whether or not the IBV can be well characterized by statistical MOP's and on the processor memory and time requirements associated with its implementation.

## B2.5 Spectral Analysis Techniques

These techniques involve the extraction of frequency information from the IBV, usually in terms of energy bands. For example, the MOP might be the energy content of a given band or the energy ratio of the fundamental to the third harmonic. It is usually much easier to do the frequency/energy determination with lumped constant circuits which are part of signal conditioning rather than digital equivalents. These equivalents, however, can certainly be used.

For digital computer implementation, we are assuming that IBV output signals are represented as number sequences, where the sequences typically represent sample values of a time-continuous signal. There are many algorithms available for converting a time signal identified by such numbers into its frequency domain representation; however, nearly all of them require that the numbers represent values of the sampled signal taken at equal intervals of time. Furthermore, care must be taken to make the sampling interval be sufficiently small to ensure capture of all significant information in the IBV output signal. The maximum permissible size of this interval is determined by application of the sampling theorem from information theory to the particular IBV in question.

In recent years there has been much work done in the area of spectral analysis of time signals by means of digital computers. It is characterized by such descriptions as "digital filtering" and "digital signal processing," and

some of the results of this work are now available in book form.* The reader is referred to these sources for further information. It should be noted that digital spectral analysis generally requires substantial computational resources, and may be difficult to use for purposes of real-time performance verification.

*See, for example, Digital Signal Processing, L. R. Rabiner and C. M. Rader, IEEE Press, Institute of Electrical and Electronic Engineers, Inc., N.Y.

## B3.0 CLASS 2 MOP's

Class 2 MOP's are those wherein information from both input and output signals of a single IBV are utilized in MOP calculation. This class is usually resorted to when deterministic statements cannot be made about the IBV output alone.

In order to use Class 2 techniques for performance verification, it is generally necessary to know a priori, the nominal transfer function of the IBV. This transfer function may be known in either the time or the frequency domain, and for verification purposes, it is assumed to completely characterize the operation of the IBV. The knowledge of the nominal transfer function permits calculation of what the IBV output should be given the input, or (in some cases), calculation of what the input should be given the output, and thus, verification of the correct operation of the IBV is possible for any particular actual sets of input and output information. In this discussion, certain assumptions are made regarding the integrity of IBV input and output signals and regarding the linearity of IBV internal operations.

### B3.1 Inverse Transform Techniques

It can be seen from Figure B3.1 that under certain circumstances, a Class 2 MOP can be represented in the complex frequency domain as

$$\text{MOP} = \text{actual input} - \text{computed input}$$

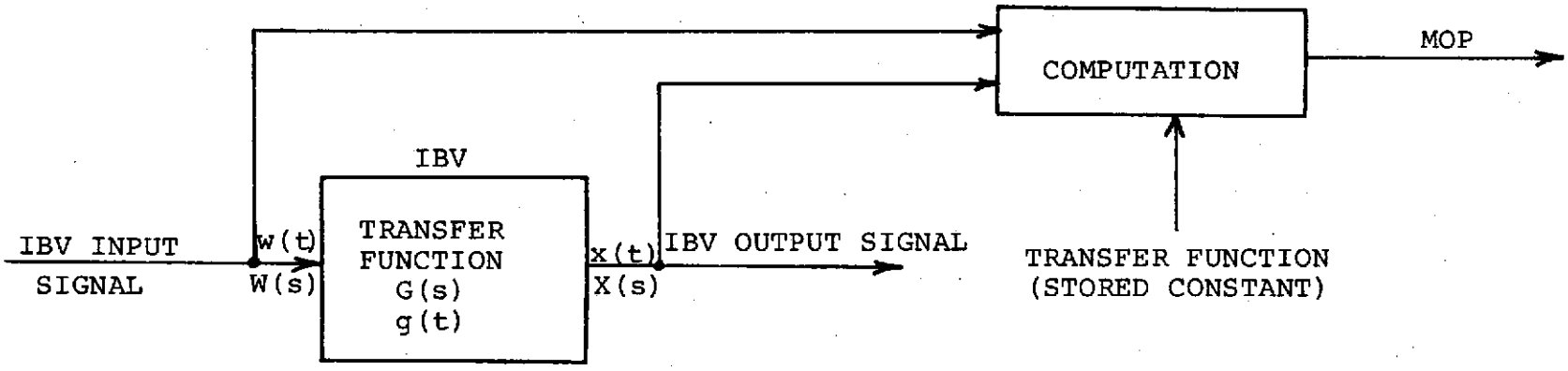
$$= W(s) - \frac{X(s)}{G(s)}$$

If the IBV has a simple transfer function, such as the case with an amplifier with constant gain A, we get

$$\text{MOP} = w(t) - \frac{1}{A}\, x(t)$$

where $g(t) = A$; therefore, $g^{-1}(t) = \frac{1}{A}$.

This is the case chosen for the example shown in Table B1.0. The example is sufficiently simple that no further explanation seems necessary. It is noteworthy that in this example, inaccuracies in MOP calculation may be significant because both $w(t)$ and $x(t)$ are represented by numbers with inherent




RELATIONSHIPS:

$W(s)$ = La Place Transform of $w(t) = \mathcal{L}(w(t))$

$X(s)$ = La Place Transform of $x(t) = \mathcal{L}(x(t))$

$G(s) = \frac{X(s)}{W(s)}$ = Transfer Function of IBV

$X(s) = W(s)G(t)$

$x(t) = \mathcal{L}^{-1}\{G(s)W(s)\}$

$$x(t) = \int_0^t g(t-\tau)w(\tau)d\tau$$

Figure B3.1. Class 2 MOP Block Diagram

measurement noise, and the resulting MOP value may contain cumulative errors due to this noise. This inaccuracy phenomenon is typical of Class 2 MOP extraction techniques. Class 1 techniques are less affected by the inaccuracy problem because only one of two numbers used in forming a difference is contaminated by noise (one of the numbers is typically an a priori-determined constant).

## B3.2 Correlation Techniques

In this technique, the actual value of the IBV impulse response function h(t) is calculated from IBV inputs and outputs and compared with the a priori-known reference value for h(t), which we call $h_R(t)$. The measure of performance is then the difference between these:

$$MOP = h_T(t) - h_R(t)$$

where $h_T(t)$ is the response function obtained with the aid of a known input signal $w_T(t)$ having duration T and satisfying certain general properties. In fact, $w_T(t)$ can take the form of a pseudorandom noise signal, generated for the purpose of stimulating the IBV appropriately.

In this case, $h_T(t)$ may be calculated from the IBV input and output signals, and is (in continuous-time representation)

$$h_T(t) = \int_0^T w_T(\tau) \times (t-\tau)d\tau$$

In other words, $h_T(t)$ is calculated as the convolution integral of w(t) and x(t) in order to obtain the desired MOP. Although a digital computer can compute a numerical approximation of $h_T(t)$ from the numbers $|X_{ijk}|$ and $|W_{ijk}|$, the operation requires substantial computational resources and may be difficult to use if real-time verification is a requirement. The particular technique just discussed is sometimes used in checking of digital data communication channels.

## B4.0 CLASS 3 MOP's

Class 3 MOP's are those computed using information from the outputs only of multiple (two or more) similar IBV's, where the similar IBV's are assumed to share the same input signals. The situation is described in Figure B4.0. This class of MOP's operates on the principle that similar IBV's, operating on the same input, should produce identical outputs. If they don't, at least one of the IBV's in the collection must have failed.

## B4.1 Compare-Two Techniques

Compare-Two techniques apply when there are only two IBV's whose individual outputs, or individual MOP's derived from outputs, are compared. A collection-IBV MOP is developed for which the value is the difference between respective individual MOP's. Essentially, if this difference is small, the collection-IBV (and also both individual IBV's) is assumed to be good; otherwise, the collection-IBV (and, therefore, one or the other of the individual IBV's) is assumed to be bad. Mathematical representations of this are shown in Table B1.0.

Note that with this technique, as with all Class 3 techniques, status resolution may not be required. Also note that with this technique, as with all other Class 3 techniques, it is not necessary to know, a priori, what the outputs of the two IBV's should be; instead, it is sufficient to know that the two outputs should be about the same with respect to their essential features, or MOP's.

The Compare-Two technique can be quite simple to implement, and it is fairly commonly used. For instance, the Compare-Two technique is currently used as part of the verification process for dual arithmetic units in some models of the IBM 370 central processing units.

The technique has, however, one potentially serious drawback. It is not possible to tell which of the two IBV's in the pair is bad. There are some limited circumstances where this information is not necessary. For most applications, this represents vital information.

## B4.2 Crosspower Spectral Analysis Techniques

Crosspower Spectral Analysis is the general name given to techniques which essentially use the Compare-Two technique on the output of two similar IBV's, but where the individual MOP's are described in the frequency domain. For example, it may be desired to compare the power spectra of

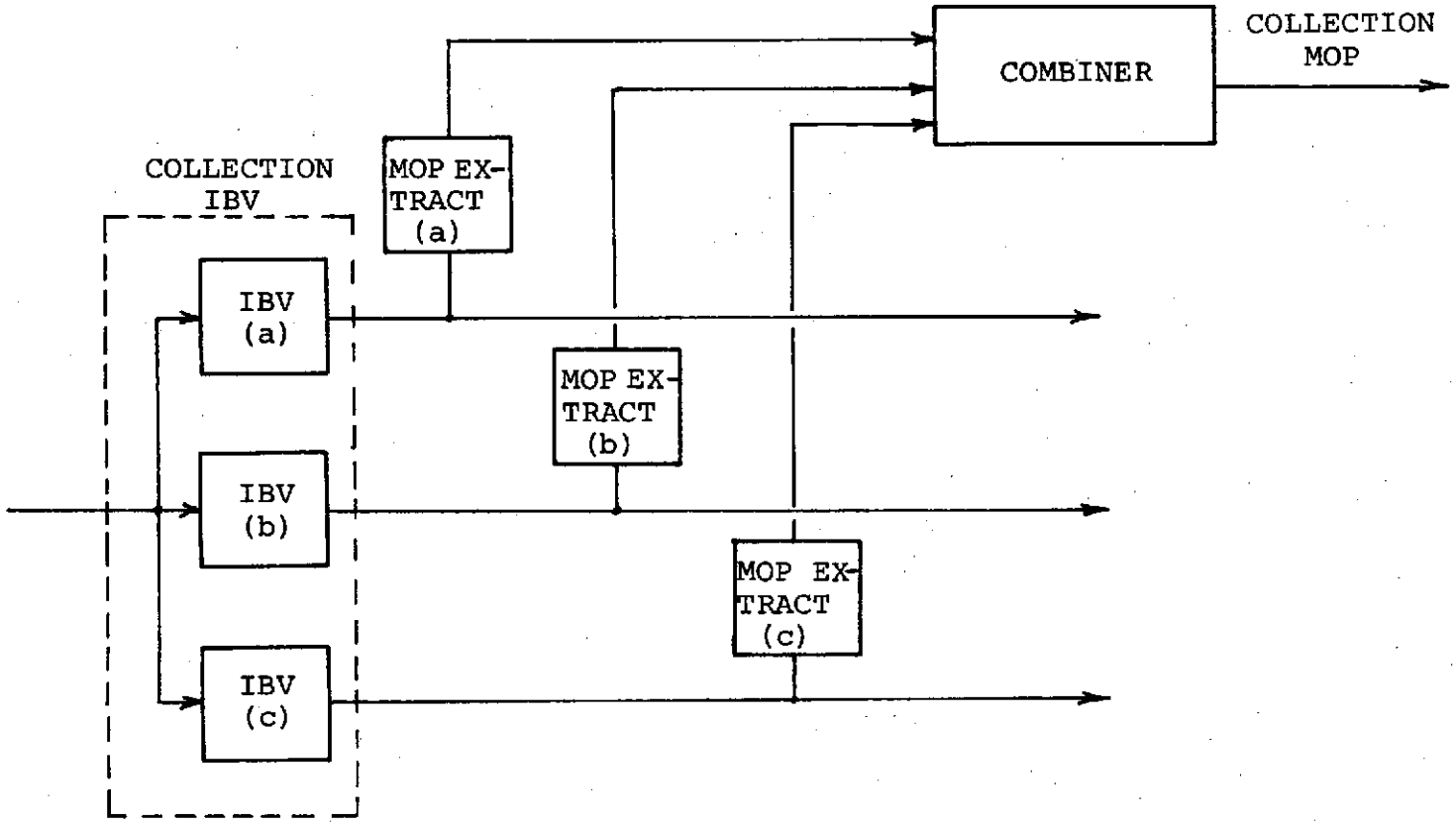



Figure B4.0. Class 3 MOP Extraction Techniques

two IBV's, which, if operating properly, should have essentially the same output power in defined frequency bands. This might be the case with dual stable oscillators, for instance. For this purpose, the coherence function defined in Table B1.0 is possibly a good choice for collection-IBV MOP.

It should be apparent from examination of the definition of coherence function that the calculation of this MOP on a general purpose digital computer is not a simple matter. The stored program necessary to realize the function will be fairly large, and program execution time will be considerable. Thus, it appears that the coherence function, or in general, any MOP of the Crosspower Spectral Analysis type, is likely to prove unsuitable for real-time performance verification purposes.

### B4.3 Voting Techniques

Voting is the only technique of Class 3 MOP extraction techniques in which the outputs of more than two IBV's are used in calculating a collection-IBV MOP. Table B1.0 shows an attempt to characterize the voting technique in mathematical terms, using the three IBV case for simplicity.

In this technique, the MOP is the mutual differences between the IBV outputs. This is often times modified to include the kind of sequential consistency described for Coding Techniques, thus, the MOP distinguishes between errors and failures. Note that the discussion of this technique has assumed a digital application. This is a limitation of this technique in the general sense. Its analog implementation is quite difficult.

Many different versions of the voting technique are possible, and the technique is sometimes used where additional decision-making reliability is judged to be worth the cost of triple (or higher) levels of redundancy. Implementation of the technique via general purpose digital computer can be quite simple, especially if the implementation computer has suitable "bit fiddling" capabilities in its instruction set. These capabilities tend to facilitate the programming of Boolean functions which define the collection-IBV MOP. This technique is being used on the GN&C computers.